On the Applicability of Short Key Asymmetric Cryptography in Low Power Wireless Sensor Networks

Von der Fakultät für MINT - Mathematik, Informatik, Physik, Elektro- und Informationstechnik
der Brandenburgischen Technischen Universität Cottbus-Senftenberg

zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften (Dr. rer. nat.)

genehmigte Dissertation

vorgelegt von

M.Sc.

Anna Sojka-Piotrowska

geboren am 26.04.1983 in Krosno Odrzanskie (Polen)

Gutachter: Prof. Dr. Peter Langendörfer
Gutachter: Prof. Dr. Antonio Skarmeta
Gutachter: Prof. Dr. Klaus Meer

Tag der mündlichen Prüfung: 30.06.2016

To my family

Contents

Extended Abstract

The growing popularity of Wireless Sensor Networks (WSN) makes the spectrum of their applications very wide. A great number of the application areas like health monitoring or military applications require a high level of security and dependability from the wireless sensor network. Solving these issues can be supported by providing cryptographic solutions into WSN applications. Since the WSNs mainly consist of low power devices, cryptographic solutions ideal for WSNs should provide computationally lightweight security mechanisms producing small data packets and ensuring confidentiality. Cryptographic mechanisms that have both these features are considered in this thesis, which main objective is the analysis of the applicability of the short key elliptic curve cryptography in WSN environments. Reduced key lengths require modification of the standard ECC security algorithms to provide authentication and also a novel solution for a cryptographic secure pseudo-random number generator. The proposed solution is based on the standard ECC, but it differs in several aspects. The main difference is that the parameters of the used elliptic curve have to be kept secret. This is due to the fact that solving the Discreet Logarithm Problem (DLP) for such short parameters can be done in short time. Additionally, using shorter parameters for the underlying elliptic curves excludes also the use of standard hash functions, what mainly influences the mechanisms for generating the digital signature. Hash functions require large input values and produce relatively large output data that is inapplicable in the shortECC environment. Thus, within this thesis a modified version of standard Elliptic Curves Digital Signature Algorithm is proposed, which does not require any hash function. The shortECC needs pseudo-random numbers in the encryption and the digital signature protocols, but since it operates on numbers that are significantly shorter than the ones used by other cryptographic approaches, pseudo-random number generators for standard approaches are not suitable for shortECC. Thus, the new pseudo-random number generator not involving any additional hardware besides the modules available on the used test platform and operating on 32-bit long integers, is proposed. The randomness of the numbers generated by the proposed algorithm

and their applicability for cryptographic purposes were evaluated using the NIST test suites. The shortECC approach was also subjected to cryptanalysis in order to proof its security and determine the circumstances and constraints for its application.

Kurzfassung

Die große Popularität drahtloser Sensornetzwerke (Englisch: Wireless Sensor Networks WSN) ist vor allem auf deren großes Anwendungsspektrum zurückzuführen. Viele Anwendungsbereiche wie die Telemedizin zur Patientenüberwachung oder auch Anwendungen für das Militär haben jedoch sehr hohe Anforderungen an die Sicherheit und Zuverlässigkeit solcher drahtlosen Netzwerke. Kryptographische Verfahren können helfen diese Anforderungen zu erfüllen. Drahtlose Sensornetzwerke bestehen allerdings oft aus energielimitierten Geräten. Entsprechend sind kryptographische Verfahren gefordert, die den Anforderungen an eine hohe Sicherheit und Zuverlässigkeit bei niedrigem Energieverbrauch genügen, d.h. leichtgewichtige kryptografische Operationen sowie die Übertragung kleiner Datenmengen. Der Fokus dieser wissenschaftlichen Arbeit liegt auf der Untersuchung von Ansätzen, die beiden Eigenschaften erfüllen. Hauptziel dabei ist die Analyse der Anwendbarkeit kryptografischer Verfahren in drahtlosen Sensornetzwerken, die auf elliptischen Kurven mit kurzen Schlüsseln basieren (shortECC).Verkürzte Schlüssellängen erfordern dabei Modifikationen der Standardalgorithmen für Kryptografie mit Elliptischen Kurven (Englisch: Elliptic Curve Cryprography ECC), um Authentifikation zu gewährleisten sowie auch eine neue Lösung für den kryptografischen Pseudozufallszahlengenerator bereitzustellen. Die vorgeschlagene Lösung (shortECC) basiert auf dem Standard ECC, unterscheidet sich aber in einigen wesentlichen Aspekten: Der Hauptunterschied liegt in der Geheimhaltung verschiedener Parameter, die im Standard ECC in der öffentlichen Domäne vorkommen. Das ist vor allem dadurch begründet, dass die Lösung des diskreten Logarithmusproblems (Englisch: Discreet Logarithm Problem DLP) mit solchen kurzen Schlüsseln für den Standard ECC sehr einfach wäre. Zudem schließt für die vorgesehene Schlüssellänge von 32-bit beziehungsweise 64-bit die Verwendung der Standard Hash-Funktionen aus. Hash-Funktionen nehmen große Daten als Eingang und produzieren Werte, die für den Einsatz in der shortECC Umgebung zu groß sind. Dies beeinflusst hauptsächlich die Algorithmen für die Generierung von Digitalen Signaturen. Im Rahmen dieser Arbeit wurde ein Algorithmus vorgestellt und evaluiert, der auf dem Standard-Elliptische Kurven Digital

Signature Algorithm (ECDSA) basiert und keine Hash-Funktion erfordert. Bei den shortECC Algorithmen werden Zufallszahlen für die Verschlüsselung und das Generieren einer Digitalen Signatur benutzt. Da aber die Standardansätze von kryptografischen Zufallszahlgeneratoren für die kurzen Zahlen nicht immer geeignet sind, wurde eine neue und sehr effiziente Lösung für einen sicheren Pseudozufallszahlengenerator vorgestellt und evaluiert. Der vorgeschlagene Generator verwendet keine zusätzliche Hardwaremodule, kann aber verschiedene Hardwaremodule zum Initialisieren nutzen. Die Qualität der generierten Zufallszahlen für kryptografische Zwecke wurde mit der NIST Test Suite mit einem sehr positiven Ergebnis evaluiert. Der vorgeschlagene shortECC-Ansatz wurde auch mit kryptoanalytischen Methoden untersucht, um die angebotene Sicherheit aber auch die Schwachpunkte und Umstände, in welchen diese entstehen, zu ermitteln.

Chapter 1

Introduction

With the increasing popularity of Wireless Sensor Networks (WSNs) the spectrum of their applications becomes very broad. And taking into account all the variations referred in the literature as Wireless Sensor and Actor Networks (WSAN), Cyber-Physical Systems (CPS) or Internet of Things (IoT) the possible applications are limited only by the imagination of the developers. A great number of the application areas, like health monitoring, homeland security or prediction of environmental threats require a high level of security and dependability from the sensor network. Thus, since the WSNs are mainly about the data, the data exchange needs to be performed securely, reliably and without fails. Solving these issues can be supported by providing cryptographic solutions into WSN applications.

The WSNs mainly consist of energy constrained and, as a result, low power devices. This causes that the application development for WSNs is not a trivial task. Such applications need to be very sensitive when it comes to energy consumed for processing of data, and also for exchanging the data within the network [47]. Both these aspects are critical and none of them can be neglected. Hence, cryptographic solutions for WSNs need to be defined or chosen carefully and that with respect to the computational effort they require, as well as to the amount of data (or data overhead) they produce.

Cryptographic approaches based on elliptic curves belong to the class of public key cryptography. One of the reasons behind the popularity of the Elliptic Curve Cryptography (ECC) is the shorter length of the keys, compared to the public key cryptography solutions based on modular arithmetic, such as the RSA approach. According to the NIST, the same security level is provided by elliptic curves over 224-bit long prime fields and by 2048-bit long factoring modulus used in cryptography based on the modular arithmetic (RSA) [3].

ECC provides a diversity of solutions for satisfying the security requirements

of the applications. The following examples are just an excerpt.

- confidentiality - Elliptic Curves Integrated Encryption Scheme or Elliptic Curve ElGamal Encryption, (both briefly introduced in Chapter 3)
- authentication and non-repudiation - Elliptic Curve Digital Signature Algorithm, (see Chapter 3)
- secure agreement of the secret keys between two (or more) parties - Elliptic Curve Diffie-Hellman Protocol, (see Chapter 3)

For the cryptography based on elliptic curves it is recommended to use prime fields with order not smaller than a 224-bit long number [3]. Although applying such elliptic curves is feasible for WSN, the computations performed on 224-bit long numbers cause large energy expenditures and take too much time if applied frequently [47] - this burden can reduce the possible sleep time and as a result shorten the lifetime of the sensor nodes. Additionally, there are applications, where the data to be secured is much shorter than the public key cryptography parameters, if the sensor measurements are stored in 32-bit long variables and each of them needs to be secured separately. Using the standard public key cryptography with a key-size that is considered to be secure, causes in such cases an enormous extension of the transmitted packets, and thus increase of the amount of energy consumed while exchanging the data. The solution ideal for WSNs should provide computationally lightweight security mechanisms that, on one hand, produce small data packets and, on the other hand, provide mechanisms ensuring confidentiality and authentication. ECC based mechanisms that have both these features are considered in this thesis. In rest of this thesis the ECC solutions that are using key lengths as recommended by NIST, will be referred to as the *standard ECC*, while the solutions with reduced key lengths will be referred to as the *shortECC*.

1.1 Objectives and Contributions

The main objective of this thesis is the analysis of the applicability of the short key elliptic curve cryptography in WSN environments. The analysis included the following steps.

- Identification of requirements and constraints that are caused by WSN and are related to the energy consumption and to the computational efficiency.
- Identification of the overall security requirements of WSN influencing further selection of cryptographic algorithms.
- Selection of the key lengths sufficient to fulfil the above mentioned requirements.

- Definition of the requirements and constraints of the cryptosystem based on the short key elliptic curve cryptography - the shortECC cryptosystem.
- Proposal for the necessary modifications in the standard ECC security algorithms.
- Comparison of the shortECC and the state of the art solutions.
- Conclusions indicating possible application scenarios for the shortECC approach.

Based on the requirements and constraints identified for the proposed shortECC cryptosystem, it was stated that it is necessary to modify the standard ECC security algorithms to provide authentication and also to propose a novel solution for a cryptographic secure pseudo-random number generator.

Lightweight Security

The proposed shortECC solution is based on the standard ECC, but it differs in several aspects. The main difference is the change in the distribution of the parameters between the public and secret domains, i.e. the parameters of the used elliptic curve have to be kept secret. This is due to the fact that solving the Discreet Logarithm Problem (DLP) for such short parameters can be done in short time. As a result, the proposed shortECC approach can be considered as hybrid solution combining the features of the secret key cryptography and the public key cryptography.

Additionally, using shorter parameters for the underlying elliptic curves excludes also the use of standard hash functions, what mainly influences the mechanisms for generating the digital signature. Hash functions require large input values and produce relatively large output data that is inapplicable in the shortECC environment. The standard ECC authentication algorithm - Elliptic Curve Digital Signature Algorithm [21] - requires a cryptographic hash function, (see Algorithm 9). Thus, within this thesis a modified version of standard Elliptic Curves Digital Signature Algorithm is proposed, which does not require any hash function. This signature scheme is described in Chapter 4, Section 4.2.2.

In order to proof the security of the proposed approach and specify the application area the cryptanalysis of shortECC was performed. The cryptanalysis considers following cases that may lead to determining the valid shortECC parameters:

- brute force attack in case when all the shortECC parameters are unknown for an adversary,
- probability of finding a proper base point in case all other shortECC parameters are known,

- finding the shortECC parameters on the basis of the eavesdropped short-ECC points in uncompressed form,
- finding the shortECC parameters on the basis of the eavesdropped short-ECC points in compressed form.

Cryptographic Pseudo-random Number Generator

Pseudo-random numbers are required in many cryptographic protocols for WSN, e.g. in the key exchange protocols [25]. The need for research on new approach was caused by the requirements of the shortECC, which needs pseudo-random numbers in the encryption and the digital signature protocols. Since the short-ECC operates on numbers that are significantly shorter than the ones used by other cryptographic approaches, pseudo-random number generators for standard approaches are not suitable for shortECC. Development of the new approach was also influenced by the shortECC assumptions to be computationally inexpensive and minimal energy consuming. Thus, the new pseudo-random number generator does not involve any additional hardware besides the modules available on the used test platform and operates on 32-bit long integers. The randomness of the numbers generated by the proposed algorithm and their applicability for cryptographic purposes were evaluated using the NIST test suites [2] [43]. The details on the novel cryptographic pseudo-random number generator are presented in Chapter 5.

1.2 Structure

The rest of this thesis is structured as follows. Chapter 2 presents the mathematical background useful to understand the theory of elliptic curves. This chapter explains also the basic definitions from the theory of cryptography required in further considerations. Chapter 3 introduces the Wireless Sensor Networks. It presents the challenges that need to be met while developing WSN applications, focusing mainly on security requirements. Further in this chapter the state of the art security algorithms and protocols for WSN are discussed. Chapter 4 presents the concept of shortECC - the public key elliptic curves cryptography with short parameters. It starts with the definition of the constraints and requirements of the environment within which shortECC may be applied. Further, it presents the security protocols being part of the shortECC cryptosystem. The chapter concludes with the evaluation of the proposed approach and comparison to the state of the art security solutions for WSN. Chapter 5 introduces the novel cryptographic pseudo-random number generator for low power devices that is based on mathematical chaos and was developed in order to fulfil the requirement of short-ECC. This chapter concludes with the evaluation of the proposed approach and

comparison to the state of the art pseudo-random number generator solutions. Chapter 6 summarizes the thesis and its main research results.

Chapter 2

Theoretical background

2.1 Finite Field Arithmetic

The finite field arithmetic serves as a basis for many cryptographic protocols. The operations performed in these fields can be used to compute and express the particular states of the cryptographic protocols. Since the elliptic curves are defined over finite fields, the computations on the elliptic curves are performed using the finite field arithmetic.

The rest of this chapter introduces the basic definitions from the abstract algebra theory leading to the theory of the elliptic curves. The abstract algebra deals with algebraic structures - sets with defined operations, where the operations are performed on finite numbers of inputs.

2.1.1 Groups

The basic algebraic structure is a group, serving also as a base for more complicated structures, e.g. finite fields. The additive group is defined as follows:

Definition 1. *An additive group $(G, \oplus)$ consists of a set G with an addition $\oplus$ on G as a binary group operation satisfying the following axioms:*

- *The group operation is associative. That is*

$$a \oplus (b \oplus c) = (a \oplus b) \oplus c, \qquad \textit{for all} \quad a, b, c \in G \tag{2.1}$$

- *There is an element $\mathcal{O} \in G$, called the identity element, such that*

$$a \oplus \mathcal{O} = \mathcal{O} \oplus a = a, \qquad \textit{for all} \quad a \in G \tag{2.2}$$

- *For each $a \in G$ there exists an element $-a \in G$, called the inverse of a, such that*

$$a \oplus -a = -a \oplus a = \mathcal{O}. \tag{2.3}$$

A group G is abelian (or commutative) if, furthermore,

- *$a \oplus b = b \oplus a$, for all $a, b \in G$, [38].*

In case of multiplicative groups, denoted G^*, the group operation is multiplication, denoted $\otimes$, the identity element is $\mathcal{I}$ and there exists inverse of a group element a, such that:

$$a \otimes a^{-1} = a^{-1} \otimes a = \mathcal{I}. \tag{2.4}$$

The groups used in cryptographic protocols have finite number of elements. Such groups are called finite groups and the number of elements in a finite group is its *order* and is denoted $\#G$. A finite group of order $n = \#G$ is denoted G_n. The concept presented in this work assumes using the groups of prime order. This assumption causes, that subgroups of G have properties, which are important when designing the cryptosystem described in this work. A subgroup is defined as follows:

Definition 2. *A non-empty subset H of group G is a subgroup of G if H is itself a group with respect to the operation of G [38].*

The Lagrange's Theorem presents the properties of the order of the subgroup.

Theorem 1. *Let G be a finite group and H be a subgroup of G. Then the order of H divides the order of G [9].*

Thus, if finite group G has a prime order n, then there are no subgroups of G having order different than n or 1. And thus, every subgroup of group G is generated by some element of G, i.e.:

Example 1. *Let $a \in G$ and $a \neq \mathcal{O}$, then the set*

$$\{a,\ a \oplus a,\ a \oplus a \oplus a,\ \ldots,\ \underbrace{a \oplus a \oplus \cdots \oplus a}_{m\ times}\} \tag{2.5}$$

is a subgroup of G generated by a and is denoted by $\langle a \rangle$.

The order of $\langle a \rangle$ is equal to the order of the element a and is defined as follows:

Definition 3. *The order of a is the smallest positive integer m, such that*

$$\underbrace{a \oplus a \oplus \cdots \oplus a}_{m\ times} = \mathcal{O}. \tag{2.6}$$

Thus, if group G has a prime order n and there are two dividers of n: 1 and n and since

$$\underbrace{a}_{1\ time} = a \neq \mathcal{O} \tag{2.7}$$

the only possible order of the subgroup $\langle a \rangle$ of G is n and the subgroup contains the following elements:

$$\{\mathcal{O},\ a,\ a \oplus a,\ a \oplus a \oplus a,\ \ldots,\ \underbrace{a \oplus a \oplus \cdots \oplus a}_{n-1\ times}\} \tag{2.8}$$

The only case when the subgroup of G can have the order equal to 1 is the case of subgroup $\langle \mathcal{O} \rangle$, containing only one element. The case when the order of the subgroup of G is equal to the order of G describes the following definition:

Definition 4. *A group G is cyclic if there is $a \in G$ such that $\langle a \rangle = G$. If such an element a exists, it is called a generator of G [9].*

In the rest of the work, only finite cyclic additive groups of prime order will be taken into consideration.

2.1.2 Finite Fields

A field consists of an additive group and of the additional operation. It is defined as follows:

Definition 5. *A field $(\mathbb{F}, +, *)$ is a set $\mathbb{F}$, with two operations: addition denoted $+$ and multiplication denoted $*$ on $\mathbb{F}$, satisfying the following axioms:*

- *$(\mathbb{F}, +)$ is an abelian group with an identity element $\mathcal{O}$.*
- *The $*$ operation is associative. That is*

$$a * (b * c) = (a * b) * c, \qquad for\ all \quad a, b, c \in \mathbb{F}. \tag{2.9}$$

- *There is a multiplicative identity denoted $\mathcal{I}$, where $\mathcal{I} \neq \mathcal{O}$, such that*

$$\mathcal{I} * a = a * \mathcal{I} = a, \qquad for\ all \quad a \in \mathbb{F}. \tag{2.10}$$

- *The multiplication operation is distributive over the addition operation. That is*

$$a*(b+c) = (a*b)+(a*c), \quad and\ (b+c)*a = (b*a)+(c*a), \qquad for\ all \quad a, b, c \in \mathbb{F}. \tag{2.11}$$

- *For all $a, b \in \mathbb{F}$ there is $a * b = b * a$.*

- *Each non-zero element $a \in \mathbb{F}$ has a multiplicative inverse, i.e., an element $b \in \mathbb{F}$ such that $a * b = \mathcal{I}$.*

The finite field is defined as follows:

Definition 6. *A finite field is a field $\mathbb{F}$ containing a finite number of elements. The order of the finite field is the number of its elements.*

Definition 7. *Let p be a prime number. The set of the integers modulo p: $\{0, 1, 2, \ldots, p-1\}$ with addition and multiplication performed modulo p, is called a finite field of order p. Such field is denoted as $\mathbb{F}_p$.*

The finite fields of prime order are called prime fields. In the rest of this work only finite fields of prime order will be considered.

The multiplicative groups of the prime field are commonly used for the cryptographic purposes. The multiplicative group $\mathbb{G}_p^*$ of the prime field $\mathbb{F}_p$ is defined as follows:

Definition 8. *Let $a \in \mathbb{F}_p$, then the multiplicative group $\mathbb{G}_p^*$ of the field $\mathbb{F}_p$ is a set: $\{a : 1 \leq a \leq p-1\}$. The group order is equal to $p-1$.*

2.2 Elliptic Curves

The elliptic curves over finite fields are used for solving many cryptographic problems, e.g., for decomposition of integers into a product of prime numbers or for primality testing. They are also used to build cryptosystems. The attractiveness of the elliptic curves is driven by the multiplicity of the finite groups (groups of points on the elliptic curves), that can be defined over prime fields and also due to the secure key lengths, which are shorter than in case of cryptosystems defined over multiplicative group of the finite field [3].

2.2.1 Elliptic Curves over Finite Prime Fields

The important issue when developing cryptosystems based on the elliptic curves is the selection of the field, which will be used as a basis for the elliptic curve. The two types of fields are considered: binary and prime finite fields and in this work only the elliptic curves over prime fields are taken into account. The elliptic curve over prime field is defined as follows:

Definition 9. *An elliptic curve E over a field $\mathbb{F}_p$, denoted as $E(\mathbb{F}_p)$, is the graph of an equation of the form*

$$E(\mathbb{F}_p) : Y^2 = X^3 + aX + b \tag{2.12}$$

where the coefficients $a, b \in \mathbb{F}_p$ are such, that for each point (X, Y) with coordinates in $\mathbb{F}_p$ the equation (2.12) is satisfied. The elliptic curve E is nonsingular, when $-16(4a^3 + 27b^2) \neq 0$, [57].

In the following, both $E(\mathbb{F}_p)$ and E notations will be used to denote the elliptic curves defined over prime fields.

Group Law

Let $E(\mathbb{F}_p)$ be an elliptic curve. For points $P = (x_1, y_1), S = (x_2, y_2) \in E$, where $P \neq \pm S$, there is $P + S = (x_3, y_3)$ where

$$x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2 \quad and \quad y_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_3) - y_1. \tag{2.13}$$

For point $P = (x_1, y_1) \in E$, where $P \neq -P$, there is $2P = (x_2, y_2)$ such that

$$x_2 = \left(\frac{3x_1^2 + a}{2y_1}\right)^2 - 2x_1 \quad and \quad y_2 = \left(\frac{3x_1^2 + a}{2y_1}\right)(x_1 - x_2) - y_1 \tag{2.14}$$

The points on the elliptic curve $E(\mathbb{F}_p)$ form an additive abelian group with following properties:

- Commutative property: for all points $P = (x_1, y_1), S = (x_2, y_2) \in E$, there is $P + S = S + P$.
- Identity element: for all points $P \in E$, there is $P + \mathcal{O} = P$.
- Inversion element: for a given point $P \in E$, there exists $-P \in E$, such that $P + (-P) = \mathcal{O}$. The point $-P$ is called the inversion (negative) of P. If $P = (x, y)$, then $-P = (x, -y)$.
- Associativity: for all points $P, S, T \in E$ there is $(P + S) + T = P + (S + T)$.

Elliptic curve point multiplication

Let $P \in E$ and $k \in \mathbb{F}_p$, then

$$kP = \underbrace{P + P + \cdots + P}_{k\ times}\ for\ k > 0 \quad and \quad 0P = \mathcal{O}. \tag{2.15}$$

The kP operation is called scalar multiplication of the point P.

The group order

The points on the elliptic curve $E(\mathbb{F}_p)$ form an additive group, where the number of points on this curve is called the order of the elliptic curve and is denoted $\#E(\mathbb{F}_p)$. The bounds for $\#E$ can be determined using the Hasse's theorem [21]:

Theorem 2. *(Hasse) Let E be an elliptic curve defined over $\mathbb{F}_p$. Then*

$$p + 1 - 2\sqrt{p} \leq \#E \leq p + 1 + 2\sqrt{p}. \tag{2.16}$$

The interval $[p + 1 - 2\sqrt{p}, p + 1 + 2\sqrt{p}]$ is called the *Hasse interval*.

The point representation - projective and affine coordinates

Each point P represented by affine coordinates (x, y) lying on the elliptic curve $E(\mathbb{F}_p)$ can be also represented using projective coordinates (X, Y, Z). For $Z \neq 0$ the affine coordinates of a point are obtained as follows:

$$(x, y) = (X/Z, Y/Z). \tag{2.17}$$

For projective coordinates the equation 2.12 can be formulated as follows:

$$E(\mathbb{F}_p) : Y^2Z = X^3 + aXZ^2 + bZ^3. \tag{2.18}$$

The inverse of $P = (X, Y, Z)$ is point $-P = (X, -Y, Z)$. Points with $Z = 0$ are all points at infinity and are denoted $\mathcal{O}$.

The projective representation of the points on an elliptic curve is useful for performing addition and doubling operations of the points. Using the projective coordinates these operations require only multiplications in $\mathbb{F}_p$, without the need for computing inversions in $\mathbb{F}_p$ - more computationally expensive operations. The only point where inversions are computed is in case when it is necessary to represent the results in affine coordinates. The elimination of the inversion causes the increase of the multiplications in $\mathbb{F}_p$, thus the purposefulness of using the projective coordinates depends elementarily on the ratio of the number of inverses to the number of the multiplications in $\mathbb{F}_p$.

Jacobian projective coordinates

Using the Jacobian projective coordinates, the point $P = (X, Y, Z) \in E$ corresponds to the affine point $(X/Z^2, Y/Z^3)$ for $Z \neq 0$. And if $Z = 0$, then it corresponds to the point at infinity $\mathcal{O}$. In Jacobian projective coordinates the point doubling is faster and the point addition is slower than for the projective coordinates [9]. The elliptic curve equation is given by:

$$E(\mathbb{F}_p) : Y^2 = X^3 + aXZ^4 + bZ^6. \tag{2.19}$$

The Jacobian projective representation of points on the elliptic curve will be applied in the rest of the work.

Point operations - Jacobian projective coordinates

According to the results presented in [21] the Jacobian projective coordinates yield the fastest point doubling and mixed Jacobian - affine coordinates yield the fastest point addition. The following algorithms present the operations of point doubling and addition for points represented in Jacobian projective coordinates.

Algorithm 1 Point doubling, Jacobian projective coordinates [4]

Input: $P = (X_1, Y_1, Z_1) \in E(\mathbb{F}_p)$.
Output: $2P = (X_2, Y_2, Z_2) \in E(\mathbb{F}_p)$.
  if $P = \mathcal{O}$ **then**
    return $\mathcal{O}$.
  end if
  $T_1 \leftarrow 3X_1^2 + aZ_1^4$.
  $Z_2 \leftarrow 2Y_1Z_1$.
  $T_2 \leftarrow 4X_1Y_1^2$.
  $X_2 \leftarrow T_1^2 - 2T_2$.
  $T_3 \leftarrow 8Y_1^4$.
  $Y_2 \leftarrow T_1(T_2 - X_2) - T_3$.
  return (X_2, Y_2, Z_2).

Point compression

For storage and transmission of the points on the elliptic curve it is of advantage if the amount of data can be reduced. In case of resource constrained devices it is important to represent these data using as few bits as possible. A point represented in affine coordinates (x, y) requires $2n$ bits to be stored, where $n = \lceil \log_2 p \rceil$. This number of bits can be reduced to $n + 1$, since for a determined x-coordinate the elliptic curve equation is a quadratic equation with an unknown y. Thus, there is only one bit needed to keep the information about the least significant bit of y-coordinate, what is enough to know which from two solutions of the quadratic equation is correct. Thus, the compressed form of the point $P = (x, y)$ includes its x-coordinate and the least significant bit f the y-coordinate. And in order to decompress the point, it is necessary to solve the quadratic equation.

First, by using the Legendre Symbol $\left(\frac{x}{p}\right)$, it has to be checked if the quadratic equation has a solution. For the prime field $\mathbb{F}_p$ the Legendre Symbol is defined as follows:

$$\left(\frac{x}{p}\right) = \begin{cases} +1 & \text{if } t^2 \equiv x \pmod{p} \text{ has a solution } t \not\equiv 0 \pmod{p} \\ -1 & \text{if } t^2 \equiv x \pmod{p} \text{ has no solution } t \\ 0 & \text{if } x \equiv 0 \pmod{p} \end{cases} \tag{2.20}$$

Algorithm 2 Point addition, Jacobian projective coordinates [4]

Input: $P = (X_1, Y_1, Z_1), Q = (X_2, Y_2, Z_2) \in E(\mathbb{F}_p)$, such that $P \neq \pm Q$.
Output: $P + Q = (X_3, Y_3, Z_3) \in E(\mathbb{F}_p)$.
  if $P = \mathcal{O}$ **and** $Q = \mathcal{O}$ **then**
    return $\mathcal{O}$.
  end if
  if $P = \mathcal{O}$ **then**
    return Q.
  end if
  if $Q = \mathcal{O}$ **then**
    return P.
  end if
  $T_1 \leftarrow X_1 Z_2^2$.
  $T_2 \leftarrow X_2 Z_1^2$.
  $T_3 \leftarrow T_1 - T_2$.
  $T_4 \leftarrow Y_1 Z_2^3$.
  $T_5 \leftarrow Y_2 Z_1^3$.
  $T_6 \leftarrow T_4 - T_5$.
  $T_7 \leftarrow T_1 + T_2$.
  $T_8 \leftarrow T_4 + T_5$.
  $Z_3 \leftarrow Z_1 Z_2 T_3$.
  $X_3 \leftarrow T_6^2 - T_7 T_3^2$.
  $T_9 \leftarrow T_7 T_3^2 - 2X_3$.
  $Y_3 \leftarrow (T_9 T_6 - T_8 T_3^3)/2$.
  return (X_3, Y_3, Z_3).

Solving the quadratic equation can be performed by using the Tonelli-Shanks algorithm [8].

2.3 Cryptography

A term cryptography relates to the techniques ensuring security of information while it is being sent or stored. Security of information can be achieved by providing mechanisms for:

- Data confidentiality - the information can be accessed only by entitled entities.
- Data integrity - the information cannot be manipulated while it is being sent.
- Authentication - the sender/owner of the information can be identified.
- Data availability - the information can be accessed when it is needed.

- Non-repudiation - the entity cannot deny sending the information.

2.3.1 Cryptographic Primitives

Plaintext

A plaintext is an information in its original, by any cryptographic technique unchanged form.

Ciphertext

A ciphertext is a text, which was obtained by encoding plaintext using some cryptographic technique.

Encryption

An encryption is an operation that encodes plaintext into ciphertext.

Decryption

A decryption operation allows to obtain the plaintext from the ciphertext.

Digital Signature

A digital signature is a set of cryptographic techniques preventing the data from being modified or tampered. It provides authentication of the data sender, data integrity and non-repudiation.

Cryptographic Keys

A cryptographic key is an information that serves as a parameter in cryptographic operations (e.g. encryption, digital signature). All the keys available for some cryptographic operation build a set known as the key space and denoted $\mathcal{K}$. There is an important fact concerning the size of the key space [9]:
A necessary, but usually not sufficient condition for an encryption scheme to be secure is that the key space has to be large enough to preclude the exhaustive search.

Definition 10. *Let $\mathcal{K}$ be the key space for a set of encryption transformations. A sequence of symbols $e_1e_2e_3 \dots e_i \in \mathcal{K}$, is called a keystream [9].*

Cryptographic Nonce

Cryptographic nonce is a number that can be used only once during the cryptographic operation. Using the nonce more than once could cause threats in security protocols.

Initialization Vector

Initialization Vector (IV) is a number of fixed length. Some cryptographic protocols require that the number is random or pseudo-random other ones require only non-repetitiveness. It is used as an input value in ciphering algorithms and helps to avoid the situation when the same plaintext after encryption results in the same ciphertext.

Cryptographic Hash Functions

A hash function is the function subordinating a number of arbitrary size to a short one, having determined length. The hash functions used for cryptographic purposes need to fulfil the following requirements:

- Collision resistance - it should be practically impossible to generate the same hash value for two different inputs.
- Pre-image resistance - for the hash values h it should be practically impossible to find message m such that $hash(m) = h$.
- One-wayness - it should be impossible to compute an input message on the basis of hash value.

Message Authentication Code

Message Authentication Code (MAC) is in cryptography a one-way function providing a code that ensures the integrity and authenticity of the message. A keyed-Hash Message Authentication Code (HMAC) is an example of MAC including the secret key in each operation and using cryptographic hash function as secure one-way function. The proper HMAC can be computed only by the entity knowing the secret key. Adding the secret key ensures the authenticity of data, which can be verified only by other entity knowing the secret key.

Cipher-based Message Authentication Code

Cipher-based Message Authentication Code (CMAC) is an algorithm based on block cipher (e.g. AES) providing the authenticity and integrity of data

2.3.2 Symmetric Key Cryptography

In the symmetric key cryptography a single and the same key is used for both encryption of a plaintext and decryption of the ciphertext. The key is kept secret by all entities taking part in the information exchange. There are two types of ciphers used in symmetric key cryptography: stream and block ciphers. The stream cipher is a symmetric algorithm which ciphers each bit of the plaintext

separately by combining it with the respective bit of the keystream (symmetric key). The bits of the plaintext and the bits of the keystream are inputs of the exclusive disjunction (XOR) operation returning 1 if only one of the inputs is equal to 1. The result of the XOR operation is the ciphertext. Decryption is done by performing the XOR operation for the ciphertext and keystream. Another type of cipher - the block cipher - performs the encryption operation on blocks, which are obtained by dividing the plaintext into strings of equal and usually predefined length. During the encryption, the input (plaintext) block is ciphered into an output (ciphertext) block using the encryption transformation and the key, where both the input and output have equal length. Without the corresponding deciphering transformation and the key this process is irreversible.

Block Ciphers - Modes of Operation

The following confidentiality modes can be used with underlying block cipher [14].

- Electronic Codebook.
  One of the simplest ciphering modes. In this mode of operation the same plaintext blocks result in identical ciphertext blocks. The blocks are encrypted and decrypted independently.

- Cipher Block Chaining (CBC).
  This mode uses an Initialization Vector, which is XORed with the first plaintext block and after that encrypted, what results in ciphertext block. The next plaintext blocks are XORed with the previously computed ciphertext blocks and encrypted. On the basis of CBC a cipher block chaining message authentication code (CBC-MAC) is proposed. It creates message authentication code from block cipher.

- Counter Mode (CTR).
  The CTR mode features the application of the forward cipher to a set of input blocks, called counters, to produce a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa.

Confusion and Diffusion

The terms confusion and diffusion were introduced by Shannon [53]. He suggested that provision of the confusion and diffusion in ciphering algorithms will help to obscure the statistical properties of the plaintext and thus prevent from statistical analysis attacks on ciphertext. The diffusion is responsible for rearranging the bits in the plaintext in order to omit the redundancies in the ciphertext. The role of confusion is to make the relationship between key and the plaintext as

complicated as possible. To achieve these both properties the block cipher uses a substitution-permutation network, which relates to the set of mathematical operations performed on the blocks of the plaintext and the key. The substitution-permutation network consists of substitution-boxes (S-box) and permutation-boxes (P-box). The S-box is a kind of black box with input, output and unknown content. The essence of its operation is to transform some number of input bits into output bits, where the number of output bits does not have to be equal to the number of input bits. The P-Box is used to permute and transpose bits in S-Box.

2.3.3 Asymmetric Key Cryptography

The asymmetric cryptography uses sets of two or more mathematically connected keys, where one of the keys is publicly known and this fact does not influence the security level provided by the cryptosystem using this key. The asymmetric cryptography is used for encryption and for digital signatures using two keys-private key and public key. It should be computationally infeasible to compute the private key on the basis of the public key. In the encryption schemes the public key is used for the encryption of the information and the private key is used for its decryption. The private key is known only to the recipient of the encrypted information and only she is able to decrypt it. The public key is known to anyone who wants to encrypt the information.

In digital signature schemes the authenticating party generates first the hash of the information. The hash is generated by the function, which subordinates a short message with defined size to the input information. This short message is encrypted by the authenticating party with its private key and enclosed as a digital signature to the original message. Anyone possessing the public key is able to verify the digital signature by deciphering the hash with the corresponding public key and comparing it to the hash the sender has generated from the original message.

The algorithms used in asymmetric cryptography are based on functions which are easy to compute but hard to invert [9]. The following paragraphs name examples of functions used for cryptographic purposes:

Discrete Logarithm Problem

The difficulty of finding the discrete logarithm, i.e. solving the Discrete Logarithm Problem (DLP), is a basis of the security in many cryptographic algorithms, like key agreement protocols or encryption mechanisms. The discrete logarithm of element b to the base a, where a and b are elements of the finite cyclic multiplicative group G of order n is the unique integer $c \in (0, n-1)$ such that:

$$a^c = b. \tag{2.21}$$

The DLP is defined as follows:

Definition 11. *Let $\mathbb{G}_p^*$ of order $p-1$ be the multiplicative group of the prime field $\mathbb{F}_p$ and a be the generator of this group. The issue of finding the integer x, where $0 \neq x \neq p-1$, such that for the element $b \in \mathbb{G}_p^*$ there is:*

$$a^x \equiv b \pmod{p}, \tag{2.22}$$

is called Discrete Logarithm Problem.

Elliptic Curves Discrete Logarithm Problem

The security of the cryptographic techniques based on Elliptic Curves is also based on the DLP.

Definition 12. *Let E be the elliptic curve over the finite field $\mathbb{F}_p$ and let P and $Q \in E(\mathbb{F}_p)$ be the points on the elliptic curve. Let $n = \#E(\mathbb{F}_p)$ be the order of the group of points on E and $m \in \{0, ..., n-1\}$ be a positive integer such that:*

$$Q = mP. \tag{2.23}$$

Having m and P it is easy to compute Q, but it is assumed to be hard to compute m on the basis of known Q and P. The issue of finding m is called the Elliptic Curve Discrete Logarithm Problem (ECDLP).

Integer Factorization Problem

The security of many cryptographic protocols (e.g. RSA [50]) is based on the fact that the efficient method for factorization of large composite integers is not known.

Definition 13. *The integer factorization problem is the following: given a positive integer n, find its prime factorization, that is, write: $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$ where the p_i are pairwise distinct primes and each $e_i \geq 1$*

2.3.4 Key Exchange

A key exchange is a cryptographic technique for exchanging the cryptographic keys between two entities willing to use other cryptographic methods, e.g. encryption algorithm. The key exchange problem is in sending the keys only to trusted parties and preventing unauthorized entities from obtaining the keys.

Chapter 3

State of the art

3.1 Wireless Sensor Networks

A wireless sensor network (WSN) is a network consisting of small and low cost devices with constrained computational and energy resources called sensor nodes. These sensor nodes cooperate in order to fulfil the tasks defined for the WSN. The wireless sensor networks are applied in diverse environments and for diverse purposes. The WSN can be used to monitor some phenomena and to deliver the sensor data to the appropriate place, e.g. sink. This cooperation is possible due to the wireless communication between nodes, which are equipped with radio transceivers and sensing or acting means. Such WSNs can be used to support the early detection of earthquakes, landslides or floods by performing the measurements of the identified phenomena and delivering the sensed data to the control centres. They also improve people's quality of life. Health monitoring supported by wearable and implantable sensor nodes enables the patients to recover at home. Wireless sensor networks integrated within houses allow better control of the consumption of energy. This increases the house owners' awareness and helps them to influence the natural resources to be consumed.

3.1.1 WSN - Challenges

There are several very specific features and requirements characterizing the wireless sensor networks which make the creation of WSNs a very challenging task.

- Hardware limitations
  The wireless sensor networks are often composed of very large number of tiny devices and deployed in places, where their presence cannot be constantly monitored. The devices can be broken, wasted or even stolen, thus the costs of the exchange or replenishment of the sensor nodes should be

acceptable for the network owner. Using the wireless sensor networks for supporting daily life should not cause huge expenses for people deciding to use it, otherwise it could be hard to introduce new technologies and ideas. As mentioned above, once deployed, sensor nodes should work without the interference of its maintainer as long as possible. Since the sensor nodes are battery powered it is a crucial issue to ensure that the selection of hardware components for the sensor boards and also the development of the software running on this hardware is done carefully and with respect to the energy resources available on the nodes.

- Network architecture
  Sensor networks can be dense and also dynamic. Due to the wireless nature of the communication some WSN applications, e.g., military ones, require that the sensor networks self-organize themselves after deployment and that the network is able to self-maintain. Self-maintenance allows adjustments in the working of the hardware and software components, in order to achieve the goals. There are important aspects that need to be constantly monitored and assured in the network, e.g., that the clocks on the nodes are synchronized in some defined periods of time. If it is also necessary for the application, each node has to know its (relative) location and the nodes being currently in its one-hop neighbourhood. Another important issue that needs to be considered when developing the WSN is the ability of self-healing, i.e., recognition of network malfunctions and appropriate reactions, in order to restore the right state of the network. There are WSN applications, e.g., environmental monitoring that require the mobility of sensor nodes. Mobile sensor nodes can enhance the capabilities of the statically deployed WSN by improvement of the sensing area coverage or by provision of communication redundancy to improve connectivity. This causes that the data paths linking the sensor nodes and the sink are not static and the network needs solutions for creating dynamic routing paths. Adding new nodes to the existing network implies that the network needs to support scalability. Scalability in turns causes that the protocols for WSN need to adapt themselves to changing number of nodes. The WSN protocols should be responsive to changes in their environment, e.g. those influencing the connectivity. Nowadays it is desired to design the wireless sensor networks that are interoperable, i.e. that the nodes belonging to one network will be able to cooperate with nodes from other networks, creating heterogeneous networks.

- Dependability
  Many wireless sensor network applications (especially the industrial ones) are very demanding in terms of the system dependability. They require that

the network will work reliable, i.e., properly and continuously for some intended period of time. In order to ensure the reliable work, the network needs to be able to cope with faults, like radio interferences, battery exhaustions or de-synchronization. Another issue is the service availability - requiring that the service provided by the wireless sensor network is not affected by the faults of any of its components (e.g. single sensor nodes or clusters of nodes) and will be operational when needed. In case when a failure appears, it is very important that the system will be restored or retained to the given condition within a defined period of time. The measure of the system restoring rapidity and the system restoring ease is known as its maintainability. Another important issue influencing the overall system dependability, is that it shall be impossible for unauthorized entities to have an access to the information that the system is using, i.e., the confidentiality of the information has to be ensured. The information used by the system must not be manipulated by any means, i.e., its integrity has to be preserved. These two requirements - confidentiality and integrity - together with availability are the most important information security goals and will be described in the following subsection.

3.1.2 WSN - Security Requirements

Most of the WSN applications operate in hostile environments and are exposed to thefts or intentional alterations of data. Also the hardware is threatened with damages, tampering or theft. Thus, provision of means solving the security issues in such resource constrained and highly distributed environment, as the wireless sensor networks are, is an important and challenging issue. There are following security issues related to WSN world:

- Data Confidentiality.
  Confidentiality ensures that any data communicated within the wireless sensor network is accessed and processed only by authorized entities - sensor nodes, cluster heads, sink or network maintainer. It is important, that the data to be sent or stored is encrypted.

- Data Availability.
  The data availability means that the data which is processed or stored is obtainable and ready for use on demand by the authorized users. It is thus important to ensure the communication between devices and also that the devices containing the data or being on the transmission path will be present and active when needed.

- Data Integrity.
  This requirement refers to the assurance that the data will have the form

intended by the issuer during its transmission, processing and storage. It has to be ensured, that the data transmitted from the source to the recipient will not be altered on the way.

- Authentication.
  It has to be ensured that the communicating node is the one it claims to be and the node receiving the data packet will be able to verify the sender.

- Non-repudiation.
  Non-repudiation ensures that the sender of a data cannot later deny that she is sender of this data, and the recipient of the data cannot deny that she has received this data.

- Data Freshness.
  This requirement refers to the assurance that the data which is sent is the most recent one, i.e., it should not be possible for an adversary to send outdated messages.

- Authorization and Access Control.
  It is required to determine the resources an entity is allowed to access and the operations the entity can perform on these resources. This is done by access control mechanisms on the basis of policies defining what resources can be used/accessed by the particular entities. Granting the entitlement to access and use the resources is called authorization.

- Forward secrecy.
  It must not be possible that a node which does not belong to the network any more is able to read the messages sent within the network.

- Backward secrecy.
  It must not be possible that a new node in the network is able to get access to the previously sent messages.

3.1.3 Attacks on Wireless Sensor Networks

Due to the fact that the nodes of a WSN may be deployed in unprotected areas it is possible that the network will be attacked. The type of the attack is dependent on the capabilities of the adversary, on her location and on her access level. If the attacker can compromise only sensor nodes she is called a mote-class attacker. Such an attacker is able to attack low-energy sensor nodes, but is not capable to perform strong attacks. Another type of the attacker is a laptop-class attacker using devices with stronger computational and energy resources than the sensor nodes are equipped with. Further, the adversary can be located inside or outside the network and can perform passive or active attacks. The following types of attacks can be distinguished in the WSN world:

- Man in the Middle attack.
  An adversary pretends to be a network member and controls the communication between the nodes. She is able to steal or even to manipulate the data. An example of such an attack is the exchange of the authorized public key used for encryption of data by one provided by the adversary. After that the adversary is able to decrypt the exchanged data.

- Node replication attack.
  An adversary can get access to the security parameters, like secret keys for example, by physically capturing a node being part of the network. After that she can prepare a malicious node having the captured security credentials and node ID, insert it into the network and induce other nodes from the network to accept the malicious node as the legitimate member of the network. The malicious node is equipped with the ID of some node being part of the network. The replicated node can drop, reroute or change the packets it gets. The malicious node can also generate false and misleading data packets.

- Spoofing, altering, replying the routing information.
  The target of this attack is the routing information exchanged by the nodes. By spoofing, altering or replaying this information the attacker can create new routing paths, change the lengths of the existing ones or generate false error messages.

- Wormhole attack.
  In this attack the malicious nodes are placed in different parts of the network. The node from one part of the network receives the messages over an alternative low latency link and sends them to the different part of the network. This causes the routing disruption (false routes), the routing race conditions, changes in the network topology and changes in the normal message flow.

- Eavesdropping, gathering, stealing the data.
  In this attack the malicious node accesses the content of the communication by listening to the message transmissions in the broadcasting wireless medium. This attack violates the data confidentiality and privacy.

- Collisions.
  In this attack the data is transmitted continuously by an adversary to cause collisions in the network, to trigger retransmissions. Collisions can also be caused by sending altered data, which due to the incorrect MAC cannot be received by the recipient. The altered packets are sent again what also causes collisions.

- Exhaustion of the resources of the sensor nodes.
  The power resources of the sensor nodes are exhausted by forcing the nodes to waste energy on pointless operations, like the continuous retransmissions.

- De-synchronization.
  In this attack the transmission between two nodes is resynchronized. This can be done by continuous sending of fake messages containing outdated sequence numbers or flags. This causes that the nodes need to synchronize by retransmitting the missed frames what depletes their resources leading to the exhaustion.

- Jamming.
  The jamming attack is the deliberate transmission of radio signals in order to disrupt the information. Jamming attack causes temporary or permanent suspension in reception and transmission of the packets by the jammed sensor node.

- Physical attack.
  In this attack the adversary performs the attacks on the devices. By tampering the sensor nodes it is possible to extract the security parameters (secret keys), or the information about the network (from the source code for example). The adversary can change the source code in order to get access to the network or replace the nodes.

- Selective forwarding/Black Hole Attack.
  It this type of attack the malicious nodes do not forward or even drop the messages. This causes data loss and may even disrupt the whole network. The case when the malicious node refuses to forward any message it gets is called a black hole attack. In this attack the neighbouring nodes have to choose an alternate path for transmitting the data.

- Sybil attack.
  In this type of attack a malicious node can present multiple identities to the other nodes in the network. This attack influences the redundancy mechanisms of distributed storage or multipath routing and data aggregation mechanisms.

- Sink-hole attack.
  In this type of attack, the adversary lures the traffic from the nearest area of the compromised node. In this case the adversary can manipulate the data packets, can modify the routing information, can fabricate and drop the messages. This attack can trigger other attacks, like selective forwarding.

- Hello flood attack.
  Some of the routing protocols for WSN require sending hello messages by

the nodes in order to discover their neighbouring nodes. In this type of attack the laptop-class attacker broadcasts a hello message to the neighbouring nodes assuring them that she is their neighbour and wants to start a communication with them. Since the attacker uses a high-power wireless link, she can reach every node in the network.

- Acknowledgement spoofing.
  Some of the routing protocols for wireless sensor networks require sending the acknowledgements. The adversary spoofs the link layer acknowledgements in order to inform the communicating nodes that some weak link is strong and assures reliable communication or that some disabled node is active.

3.2 Security Solutions for Wireless Sensor Networks

3.2.1 Symmetric Key Cryptography

This section presents the most important symmetric key cryptography solutions used in WSN.

RC4

The popular stream cipher *RC4* was designed by Rivest in 1987. The encryption and decryption operations are performed by using a keystream. This keystream is generated using a pseudo-random number generator for which the seed is obtained in the key setup algorithm. As presented by Rivest in [49] there are known attacks on the *RC4*, thus in 2014 the author proposed the improved variant of *RC4*, called *Spritz*. The method is still new and requires deep cryptanalysis.

Data Encryption Standard

The Data Encryption Standard (DES)is a symmetric block cipher and was used as an encryption standard in USA until 2001. Due to the short key length, i.e. 56 bits, it is not recommended currently for most of the applications. In 1999 the DES key was broken and it took 22 hours and 15 minutes using 1856 chips which were coordinated by one PC.

Advanced Encryption Standard

The Advanced Encryption Standard (AES) is a symmetric block cipher accepted as an encryption standard [42] by American federal agency - National Institute of Standards and Technology (NIST). The cipher is based on the Rijndael cipher [12] proposed by Joan Daemen and Vincent Rijmen. Originally the Rijndael cipher supports several block and key lengths, which can be independently chosen as

128-, 192- or 256-bit ones. For the AES standard the NIST allowed using only 128-bit long blocks and choosing from three key lengths, i.e. 128, 192 and 256 bits. The algorithm performs 10 (for 128-bit long key), 12 (for 192-bit long key) or 14 (for 256-bit long key) ciphering rounds. Each round consists of operations performed within substitution-boxes and permutation-boxes.

Skipjack

The Skipjack algorithm was invented by American National Security Agency (NSA) in 1998 [44]. It is a Feistel cipher (Feistel network) structure and uses 80-bit long symmetric key and 64-bit long blocks and performs 32 ciphering rounds. The Feistel network allows for encryption and decryption using the same function, which do not have to be one-way function.

3.2.2 Key Management Schemes

There are three types of key management schemes applicable to wireless sensor networks.

First one is the *Trusted Server Scheme* depending on the trusted party (base station) used for key agreement. This scheme uses symmetric cryptography for data encryption and the base station is responsible for establishing the key agreement between two communicating entities. Each sensor node needs to store only one secret key, but it has to communicate with the base station each time it wants to establish communication with another node. Another type of the key management scheme is the *Self Enforcing Scheme* using public key cryptography, e.g. public key certificates, for key management. The scheme is computationally expensive, because the public key cryptography requires rather complex computations.

The third type of key management scheme is *Key Pre-distribution* where the key information is distributed among sensor nodes prior to deployment.

Diffie-Hellman key exchange protocol

The Diffie-Hellman key agreement protocol [13] was designed by Witfield Diffie and Martin Hellman in 1976. Its strength is based on the difficulty of computing the discrete logarithm problem in finite fields. The key agreed using this algorithm can be used for encryption of the communication data. It allows for secure key agreement using public communication channels even though there is an entity eavesdropping on this operation, but in turn it does not prevent from man in the middle attacks. The protocol is not constrained to be used by two parties only-the key can be agreed by an arbitrary number of entities. The basic version of the algorithm is presented by Algorithm 3.

Algorithm 3 Diffie-Helmann key agreement [38]

Summary: A and B each send the other one message over an open channel.
Result: shared secret K known to both parties A and B.

1. *One-time setup.* An appropriate prime p and generator α of G_n^*, where $2 \leq \alpha \leq n - q$ are selected and published.
2. *Protocol messages.*
A sends to B: $\alpha^x \pmod n$ (1)
B sends to A: $\alpha^y \pmod n$ (2)
3. *Protocol actions.* Perform the following actions each time the shared key is required.
(a) A chooses a random secret x, such that $1 \leq x \leq n-1$, and sends B message (1).
(b) B chooses a random secret y, such that $1 \leq y \leq n-1$, and sends A message (2).
(c) B receives α^x and computes the shared key as $K = (\alpha^x)^y \pmod n$.
(d) A receives α^y and computes the shared key as $K = (\alpha^y)^x \pmod n$.

Elliptic Curves Diffie-Hellman protocol

In 1985 Miller [39] and Koblitz [25] suggested independently to use Elliptic Curves in cryptography. Miller proposed an analogue of the Diffie-Helmann protocol over group of points on the elliptic curve stating that it is 20% faster than the protocol based on the G_n^*. The secret key can be derived form the point K, taking for

Algorithm 4 Elliptic Curves Diffie-Helmann shared secret agreement

Summary: A and B each send the other one message over an open channel.
Result: shared secret K known to both parties A and B.

1. *One-time setup.* A and B agree on elliptic curve $E(\mathbb{F}_p)$ and choose point $P \in E(\mathbb{F}_p)$ such that, the subgroup generated by P has large order.
2. *Protocol messages.*
A sends to B: aP (1)
B sends to A: bP (2)
3. *Protocol actions.* Perform the following actions each time the shared secret is required.
(a) A chooses a random secret $a \in (0, \#E(\mathbb{F}_p))$, and sends B message (1).
(b) B chooses a random secret $b \in (0, \#E(\mathbb{F}_p))$, and sends A message (2).
(c) B receives a and computes the shared secret as $K = baP$.
(d) A receives b and computes the shared secret as $K = abP$.

example some defined number of bits from the x-coordinate of this point. The security of this approach is based on the hardness of the Elliptic Curves Discrete Logarithm Problem (ECDLP). The ECDLP is presented by Algorithm 4.

3.2.3 Asymmetric Key Cryptography

This section presents the most important asymmetric cryptographic approaches.

RSA

RSA [50] is one of the first and most commonly used asymmetric key cryptography approaches. It was designed in 1977 by Rivest, Shamir and Adleman. The algorithm can be used for both encryption and generating digital signatures. In [50] the idea behind the RSA algorithm is explained as follows:
A message is encrypted by representing it as a number M, raising M to a publicly specified power e, and then taking the remainder when the result is divided by the publicly specified product, n, of two large secret prime numbers p and q. Decryption is similar; only a different, secret, power d is used, where

$$e \cdot d \equiv 1 \mod (p-1) \cdot (q-1). \tag{3.1}$$

The security of the system rests in part on the difficulty of factoring the published divisor, n.

ElGamal

The ElGamal algorithm [16], next to RSA, is one of the most popular public key cryptography methods used for both encryption and digital signatures. Its security is based on the difficulty of solving the discrete logarithm over the finite field of prime order. It was proposed by Taher ElGamal in 1985 and it implements the Diffie-Hellman key exchange algorithm to encrypt and decrypt messages. In order to encrypt a message the public private key pair is required. The procedure of generation of key pair is presented by Algorithm 5.

Algorithm 5 Key generation for ElGamal encryption [38]

Summary: Each entity A creates a public key and a corresponding private key.

Each entity A should do the following:
1. Generate a large random prime n and a generator α of the multiplicative group G_n^* of the integers modulo n.
2. Select a random integer a, such that $1 \leq a \leq n-2$, and compute $\alpha^a \pmod{n}$

3. A's public key is (n, α, α^a); A's private key is a.

The encryption and decryption procedures are presented by Algorithm 6.

Elliptic Curves ElGamal

In 1985 Koblitz proposed an analog of the ElGamal system [25]. The decision was justified by the results of the research on the Elliptic Curve Discrete Logarithm

Algorithm 6 ElGamal public key encryption and decryption[38]

Summary: B encrypts a message m for A, which A decrypts

1. *Encryption.* B should do the following:
(a) Obtain A's authentic public key (n, α, α^a).
(b) Represent the message as an integer m in the range $0, 1, 2, ..., n-1$.
(c) Select a random integer k, such that $1 \leq k \leq n-2$.
(d) Compute $\gamma = \alpha^k \pmod{n}$ and $\delta = m \cdot (\alpha^a)^k \pmod{n}$.
(e) Send the ciphertext $c = (\gamma, \delta)$ to A.
2. Decryption. To recover plaintext m from c, A should do the following:
(a) Use the private key a to compute $\gamma^{n-1-a} \pmod{n}$ (note: $\gamma^{n-1-a} = \gamma^{-a} = \alpha^{-ak}$).
(b) Recover m by computing $(\gamma^{-a})\delta \pmod{n}$

Problem showing that there is no efficient method for solving it. The methods for key generation and for encryption and decryption are presented by Algorithm 7 and 8 respectively.

Algorithm 7 Key generation for Elliptic Curves ElGamal encryption

Summary: Each entity agrees on elliptic curve $E(\mathbb{F}_p)$ and chosen point $P \in E(\mathbb{F}_p)$ such that, the subgroup generated by P has large order.
Each entity A should do the following:
1. Select a random integer a and compute aP
2. A's public key is point aP; A's private key is a.

Algorithm 8 Elliptic Curves ElGamal public key encryption and decryption

Summary: There is an agreed elliptic curve $E(\mathbb{F}_p)$ and a point $P \in E(\mathbb{F}_p)$ such that, the subgroup generated by P has large order.
B encrypts a message m for A, which A decrypts
1. *Encryption.* B should do the following:
(a) Obtain A's authentic public key aP.
(b) Represent the message as point M on the elliptic curve E.
(c) Select a random integer k.
(d) Compute $M_1 = kP$ and $M_2 = M + kaP$.
(e) Send the ciphertext M_1, M_2 to A.
2. Decryption. To recover plaintext M from M_1 and M_2, A should do the following:
(a) Use the private key a to compute aM_1.
(b) Recover M by computing $M_2 - aM_1$

Elliptic Curves Integrated Encryption Scheme

This approach is a hybrid encryption method based on the elliptic curves cryptography and on symmetric encryption algorithm [34]. The method requires the

following functions in order to encrypt/decrypt messages:

- key agreement protocol
- key derivation function
- message authentication code
- symmetric encryption scheme
- hash function
- elliptic curves ElGamal encryption and decryption scheme
- method for random numbers generation.

3.2.4 Data authentication schemes

Most of the applications in wireless sensor networks require assurance that the data sent within the network originates from the correct sender. For this purpose the mechanisms for data authentication are used. These mechanisms comprise both symmetric key cryptographic approaches and public key cryptography ones. But, the first ones allow for authentication of messages sent between two parties sharing the secret key. The second ones allow for authenticated broadcast, i.e. the authenticity of data can be verified by all the parties knowing the public key of the party which authenticated the data before sending. There are following approaches that can be used for authentication purposes.

Digital signatures with appendix

In this type of digital signatures the original message that needs to be signed is required in the verification phase and the hash functions are used in order to produce the signature. One of the most popular ECC algorithms used for digital signatures with appendix is the Elliptic Curve Digital Signature Algorithm [21]. It requires a cryptographic secure hash function and its security is based on the ECDLP. The methods for signature generation and for its verification are presented by Algorithm 9 and 10 respectively.

Digital signatures with message recovery

In digital signatures with message recovery the verification phase does not require the original message, which in turns is recovered from the signature. Such type of digital signature is proposed in [56]. The concept adopts the idea of self-certified public keys [20] which do not require that the certificates used for authentication are separate values. In this case the certificates are parts of the public key. The user owns a public key which is derived from the signature of its private key

Algorithm 9 ECDSA signature generation [21]

Summary: Elliptic curve $E(\mathbb{F}_p)$ of order n and point $P \in E(\mathbb{F}_p)$, hash function h, message m and signer's private key d.

Result: Signature (r, s).

1. Select $k \in [1, n-1]$.
2. Compute $kP = (x_1, y_1)$.
3. Compute $r = x_1 \pmod{n}$. If $r = 0$ then go to step 1.
4. Compute $e = h(m)$.
5. Compute $s = k^{-1}(e + d * r) \pmod{n}$. If $s = 0$ then go to step 1.
6. Return (r, s).

Algorithm 10 ECDSA signature verification [21]

Summary: Elliptic curve $E(\mathbb{F}_p)$ of order n and point $P \in E(\mathbb{F}_p)$, hash function h, message m, signer's public key Q and signature (r, s).

Result: Acceptance or rejection of the signature.

1. Verify that r and $s \in [1, n-1]$. If any verification fails then return REJECT SIGNATURE.
2. Compute $e = h(m)$.
3. Compute $w = s^{-1} \pmod{n}$.
4. Compute $u_1 = ew \pmod{n}$ and $u_2 = rw \pmod{n}$.
5. Compute $X = u_1 P + u_2 Q$.
6. If $X = \mathcal{O}$ then return REJECT SIGNATURE.
7. If x-coordinate of X is equal to r then return ACCEPT SIGNATURE; else return REJECT SIGNATURE.

with his identity and is signed by the trusted authority using the private key of the authority. The secret key of the user is chosen by the user and is secret for the trusted authority. The digital signature with message recovery based on self-certified public keys is divided into three phases: initialization, signature generation and message recovery. In the initialization phase the trusted authority chooses two large primes p and q and publishes $N = p*q$ additionally is selects an integer g and an one-way function $h()$ which are also public. The user U_i with the identity ID_i willing to join the system chooses its private key x_i and computes $p_i = g^{x_i} \pmod{N}$ and sends its identity and p_i to the trusted authority. The authority computes $y_i = (p_i - ID_i)^{h(ID_i)^{-1}} \pmod{N}$ and publishes its as public key of the U_i. In order to sign the message M user U_i chooses a random integer k and computes:

$$r = M * g^{-k} \pmod{N} \tag{3.2}$$

$$s = k - x_i * h(r) \tag{3.3}$$

and sends the signature (r, s) to the recipient. The verification of the signature is performed as follows:

$$M = r * g^s * (y_i^{h(ID_i)} + ID_i)^{h(r)} \pmod{N}. \tag{3.4}$$

Another approach proposing the identity based digital signature with message recovery is presented in [59]. Its security is based on the discrete logarithm problem. The scheme assumes existence of the trusted Private Key Generator (PKG) which public key takes part in the signature verification phase.

Digital signature not using hash function and resistant to forgery attacks is proposed in [7]. There are following elements needed in order to sign and verify the message M: large prime p being the order of the finite field $\mathbb{F}_p$, base element $g \in \mathbb{F}_p$, private key x_i of the signer U_i and the corresponding public key $y_i = g^{x_i} \pmod p$. In the signing procedure the user U_i perform following computations:

$$s = y_i^M \pmod p. \tag{3.5}$$

Then U_i chooses random integer $k \in \mathbb{F}_p$ and computes

$$r = M * s * g^{-k} \pmod p. \tag{3.6}$$

After that user U_i computes t, where

$$s + t \equiv x_i^{-1} * (k - r) \pmod p - 1 \tag{3.7}$$

The signature is the triple (s, r, t). In the verification phase the verifier computes:

$$M' \equiv y_i^{(s+t)} * r * g^r * s^{-1} \equiv g^{x_i(s+t)} * M * s * g^{-k} * g^r * s - 1 \equiv g^{(k-r)} * M * g^{(-k+r)} \pmod p \tag{3.8}$$

If it is true that

$$s = y_i^{M'} \tag{3.9}$$

then the signature is valid and indeed generated by the user U_i.

Authenticated encryption

Authenticated encryption is based on block ciphers but provides simultaneously data confidentiality and integrity and data authenticity by combining encryption schemes and message authentication codes. The most commonly used scheme is the one called Counter with CBC-MAC mode (CCM) [15]. It is based on symmetric key block cipher, with block size of 128-bits, such as AES algorithm. It uses a single key and combines counter mode encryption and cipher block chaining-based authentication. The CCM mode consists of two processes: generation-encryption and decryption-verification. In generation-encryption phase, CBC is applied to the payload, the associated data, and the nonce to generate a MAC. After that, the counter mode based encryption is applied to the MAC and the payload to transform them into the ciphertext. In this mode the length of the payload is extended by the size of the computed MAC. In decryption-verification phase, counter mode based decryption is applied to the ciphertext to decrypt the MAC and the corresponding payload. Then, cipher block chaining is applied to the

payload, the received associated data, and the received nonce to verify the correctness of the MAC. Successful verification assures that the payload and the associated data are provided by the issuer having the proper secret key. Another approach for authenticated encryption based on AES is the Galois/Counter Mode (AES-GCM) of operation [37]. GCM uses counter mode of operation and builds on it by adding a MAC based on hash function. It uses polynomial hashing in the binary finite field.

3.2.5 Security Protocols for WSN

LSec

In [52] the authors propose *LSec - a lightweight security protocol for wireless sensor networks*. It is a combination of the trusted server and self enforcing key management schemes. LSec supports both mobile and static environments with one or multiple Base Stations. There are two main assumptions in this system. First, the Base Station is the trusted party and it cannot be compromised. Second, the public keys of all nodes in the network are known only to the Base Station.

In this system a Key Management Module is used for storing public and secret keys of the nodes in a database. A Token Generator Module generates tokens for the requesters. The tokens are used by other communicating parties in order to authenticate the requesters. The Authorization Module is used for checking if some node is allowed to communicate with other nodes in the group. There is an Intrusion Detection System (IDS) detecting anomalies in the network. The IDS can be notified by the mobile agents installed on cluster heads about alarm situations.

LSec has three phases, authentication and authorization phase, key distribution phase and data transmission phase. There are seven types of packets used in this system: Request, Response, Init, Acknowledgement (ACK), Data, Update Group Key and Alert. Authentication and authorization are performed while exchanging the Request and Response messages using symmetric cryptography. In the key distribution phase the exchange of Init and ACK packets using the asymmetric scheme is done.

When some node A wants to communicate with node B, it has to get a token and the pubic key of node B from the Base Station first. The Base Station checks if node A is authorized and can communicate with node B. If yes, the Base Station generates the token encrypted with the secret key shared between node B and Base Station. Node A adds the token to the Init packet which is sent to node B. Node B is able to authenticate node A according to the received token. If the token is correct, node B generates the session private key, encrypts it and sends it to A. This key is used to encrypt the data packets exchanged between A and

B and it is deleted after transmission.

The cluster heads are responsible for updating the secret key for group communication periodically. LSec uses 6 keys taking 72 bytes of memory storage and introduces about 74 bytes of transmission and reception cost while exchanging the packets during authentication, authorization and key exchange.

TinySec

TinySec [24] is a lightweight and efficient security architecture for wireless sensor networks. It provides authenticity, integrity, and confidentiality of messages and also allows for in-network processing.

This scheme supports two security options: authenticated encryption (TinySec-AE) and authentication only (TinySec-Auth). In case of authenticated encryption the data payload is encrypted and authenticated using Message Authentication Code (MAC). The MAC is computed for encrypted data and the packet header. In authentication without encryption the packet is authenticated with a MAC, but the payload is not encrypted.

For encryption the Skipjack algorithm in Cipher Block Chaining (CBC) mode is used and to achieve the semantic security a special format of initialization vector (IV) is used. TinySec is implemented in nesC-the programming language of TinyOS and requires 728 bytes of RAM and 7146 bytes of program space. Sending the packets using TinySec-AE causes 10% increase of energy consumption when compared to sending a regular TinyOS packet. In case of packets using TinySec-Auth there is 3% increase of energy consumption in comparison to the regular ones.

SPINS

A set of security protocols for sensor networks is presented in [46]. It consists of two building blocks: SNEP and μTESLA. TinyOS is used as operating system. In this security approach each node shares its master secret key with the Base Station and all other keys are derived from this master key.

SNEP provides data confidentiality, semantic security, two-party data authentication, integrity and freshness. It has low communication overhead adding only 8 bytes per message. To provide semantic security without sending additional data, two counters are shared by the parties and used for the block cipher in the counter mode (CTR). The MAC is used for two-party authentication and data integrity. Independent keys are derived for encryption and MAC.

Two communicating nodes share the master key and they derive independent keys using the pseudo-random function F. There are different keys for each direction of communication.

SNEP offers semantic security, i.e., incrementation of the counter causes that the same message is encrypted differently each time. Data authentication, replay

protection, weak freshness and low communication overhead are achieved with MAC.

The μTESLA provides authentication for data broadcast. It is based on TESLA [45] which is not applicable to the sensor networks, since it uses computationally expensive digital signature algorithm. Since the authenticated broadcast requires an asymmetric mechanism, which is too expensive for sensor networks, μTESLA introduces asymmetry trough a delayed disclosure of symmetric keys. This scheme requires a time synchronization between nodes and the Base Station. Each sender generates the chain of secret keys, where every key is associated with an appropriate time interval. Key A is used to compute the Message Authentication Code of packet in time interval A and is revealed when the defined time expires. The recipient stores all the received packets and waits for disclosure of keys to check the authenticity of the packets. The receivers need to know the key disclosure schedule to eliminate forged packets. They have to be sure, that the received packets were authenticated with not revealed keys.

LEAP+

LEAP+ [60] is a key management protocol for sensor networks supporting in-network processing. Four types of keys are provided in this scheme: an individual key shared with the Base Station, a pairwise key shared by two nodes, a cluster key shared by neighbouring nodes and a global key shared by all the nodes in the network. One way key chains are used to provide a weak local broadcast authentication.

The Base Station is assumed to be a laptop class device, supplied with long-lasting power and it cannot be compromised. Every sensor node has to be able to store hundreds of bytes of keying material. The sensor nodes are not mobile.

The individual keys are preloaded into nodes prior to their deployment. These keys are generated using the pseudo-random function and a master key, which are known to the system controller.

The pairwise keys are most commonly used. There are two schemes for establishing these keys. A basic scheme assumes that the set of neighbours of a node is static and that the newly added node will discover its neighbours at the time of its deployment. Each node has a initial key loaded by the controller and derives a master key from it. A new node sends a HELLO message containing its ID to its neighbours and receives ACKs authenticated with the master keys of the neighbours. It also starts a timer to trigger initial and master key erasure. Having the initial key, the new node is able to derive the master key of its neighbours. Both nodes can now compute their pairwise key without exchanging any message. When the timer expires the new node erases the initial and the master keys and uses only pairwise keys. If a compromised node will be detected, all its neighbours will delete the keys shared with this node. The second scheme is an

extension of the basic scheme. The controller generates a chain of random initial keys, which are mapped to the different time intervals. A node deployed in time interval A receives the initial key for this interval and derives its master key for the current time interval. It also receives all the master keys for the future time intervals, but does not receive any initial key corresponding to those time intervals. In the key establishment phase the interval IDs are added to the HELLO message and the ACKs are authenticated using current master key according to the interval ID. The establishment of the cluster key is following the phase of the pairwise keys establishment. The node that wants to establish the cluster key, generates first a random key and sends it, encrypted with the pairwise keys, to each of its neighbours. The neighbours store the random key in a table and send their own cluster keys, encrypted with pairwise keys, to the new node. When new node is revoked, the neighbours generate new cluster keys.

The global key is preloaded to the nodes, but in case when a compromised node is detected, the global key is changed and securely distributed to remaining nodes. First the node revocation operation must be performed. The controller sends an authenticated broadcast message using μTESLA protocol with the information about the compromised node. The neighbours of this node delete the pairwise keys and generate new cluster keys. The distribution of the new global key assumes that a secure routing protocol is used in the network.

LEAP++

The improved variant of LEAP+ called LEAP++ is presented in [32]. A one-time master keys and shorter time intervals are used, what improves the node compromise resilience. In this scheme all the nodes are programmed to start the neighbour discovery at some time after deployment. One-time master key is used for each incremental deployment and an one-way key chain is generated from it. Each one-way key is used for a short period of time. For pairwise key establishment a node sends a HELLO messages in each time interval. These messages are verified by the neighbours and when the verifications are correct the responses are sent. The response is authenticated with a MAC, which is further verified. The authentication keys are disclosed after some time and the nodes can be verified by the neighbours. The authors propose also using short key ECDH to exchange public keys in the encrypted form under LEAP++. The master keys can be derived from two shared keys of LEAP++ and ECDH. The ECC-128 or ECC-96 can be used as lightweight cryptosystems.

A key-management scheme for distributed sensor networks

Another key management scheme for distributed sensor networks is presented in [17]. The key distribution has three phases: key pre-distribution, shared-key discovery and path-key establishment. In the first phase a large pool P of keys

is generated together with key identifiers. Then for each sensor a key ring of k keys from P is derived and loaded into the memory of the sensor nodes. Key identifiers of the key ring and the identifier of the sensor node are stored on a trusted controller node.

To ensure that any two nodes share at least one key with a probability of 0.5, there have to be 75 keys drawn from the set of 10000 keys to be on any key ring. The shared-key discovery phase is done during the initialization of the whole network, where each node discovers its neighbours. Two nodes can check if they share keys by broadcasting the identifiers of the keys they posses in the key ring. This phase builds the topology of the sensor array. If two nodes share a key, there is a link between them and all communication on that link is secured by link encryption. The third phase assigns a path-key to nodes that do not share a key but are connected by links.

When a node is compromised, the revocation message is broadcast by the controller node. This message contains the key identifiers of the key ring of the compromised node. The list is signed by the controller and the signature is unicast to each node in encrypted form. After verification of the signature, the nodes delete the keys shared with compromised node. Some links can disappear. Thus, the shared-key discovery and path-key phases are restarted to reconfigure links in the network.

In [6] a modification of the above described key management protocol is proposed. Instead of finding the shared key by the two neighbouring nodes in their key rings, the nodes need to find q common keys. This increases the resilience against node capture. These keys are then used to compute a link key by hashing all the common keys.

Message Specific Puzzle

An approach mitigating DoS attacks against both μTESLA-based and signature-based broadcast authentication is presented in [41]. The idea is to use a weak authenticator of the node, which is efficiently verifiable. Only if the weak authenticator can be verified, the node performs the expensive signature verification or packet forwarding. The mechanism for weak verification is called *message specific puzzle*. This mechanism is independent from the broadcast authentication scheme and can be seen as an additional layer of protection.

To provide a weak authentication a one-way key chain is used. The chain is generated as follows: first a random value is selected as the last key in the chain. The last but one key is generated by performing the hash function for the last key. The next key is always generated by hashing the previous key. The first key in the chain is called the commitment of the chain. Before the packets are broadcast in the network, the sender sends the commitment of the key chain to all the nodes in the network.

The authors assume, that the commitment is reliably distributed to all receivers and that, the sender is more powerful than regular sensor nodes. The receivers must keep the most recently authenticated weak authenticator and its index. The receiver is able to check if the received packet has already been previously authenticated by another receiver (replay detection) and if it is the case, this packet is discarded.

On the basis of this approach the *message specific puzzle* is developed. The idea is to use cryptographic puzzle to reduce the chances of an eavesdropper to use the observed weak authenticator for forging the broadcast packets. The broadcast message, its index and broadcast authenticator and the weak authenticator are considered as the puzzle. The attacker cannot pre-compute the puzzle as long as the key is undisclosed. For each puzzle a solution is computed. The solution has to fulfil two conditions. It has to be computed for the current key from the one-way key chain and after applying the hash function to the puzzle and its solution we get an image where the first l bits are all zeros. The l parameter is called the strength of the puzzle. The sender broadcast the message together with its puzzle. The receiver has to verify the key and if it is valid, the puzzle solution is verified.

The using of the one-way hash function causes that an attacker has to look in a whole solution space in order to find a weak authenticator. The attacker needs (on average) to check 2^l hash function operations to find a puzzle solution and using one-way key chains prevents an attacker from pre-computing the puzzle solutions.

The *message specific puzzle* brings light computational overhead on sensor nodes. To verify the weak authenticator only a few hash function operations are needed. The sender of the message has to pre-compute the one-way key chain before the deployment. The puzzles require up to $l + 6$ bits space in the packet for $l \leq 128$.

Short-lived RSA/Rabin broadcast authentication

A broadcast authentication using short-lived RSA/Rabin signatures for sensor networks is proposed in [31]. The proposed scheme is usable in case if only the authentication is important in the network. It is sufficient that the signatures are unforgeable for some defined period of time and that the new instance of chosen RSA parameters is provided to the network.

The RSA-512 can be chosen, since it requires less energy for the computations and it is proved that factoring of the RSA-512 moduli takes more than few hours. The main difficulty lies in the periodical distribution of the new parameters for the signatures. The author proposes using a kind of DoS-resistant, long interval μTESLA [46] timed one-way key chains for the distribution. The one-way and hash functions and MAC are built on the basis of a single block cipher AES-128.

The hash function is built using the Matyas-Meyer-Oseas construction. For message authentication Cipher Clock Chaining Message Authentication Code (CBC-MAC) or Cipher-based MAC (CMAC) can be used. For broadcast authentication the probabilistic Rabin signature scheme with message recovery is used.

The length of the RSA moduli should be between 512 and 768 bits. For energy efficiency a special modulus can be used, which contains information about security parameters or ID. This information is represented as high-order bits of the modulus and is embedded during the generation of the modulus. It is shown that it is highly unlikely that the number field sieve can factor this special modulus faster than a regular RSA one. To defend from DoS attacks the weak authentication [41] is used.

The lifetime of the sensor network is divided into equal intervals *T-intervals* and from these intervals a sequence of longer time intervals is built, the are called *R-intervals*. Each *R-interval* is divided into two parts: *commit interval* and *reveal interval*. A fresh RSA modulus is assigned to each *R-interval* and to each *T-interval* a key from the one-way key chain is assigned. The RSA modulus from interval $i+1$ is committed in *commit interval* i and revealed in *reveal interval* $i+1$. The fresh RSA modulus is distributed using a TESLA-like protocol. It consists of three phases: KDM-precom, KDM-commit and KDM-reveal. During the KDM-precom phase a MAC for message m and the key K are computed. For the MAC the weak authenticator w is computed. In KDM-commit phase the MAC and w are broadcast and in the KDM-reveal phase the message m and the key K are disclosed. The packets sent consist of the message, its type and the sender's ID. The type of the message allows the receivers to properly process the message and the sender's ID is used to support multiple senders.

3.3 Random Number Generators for WSN

Random or pseudo-random numbers are required by many security protocols. An inproper source of randomness can cause that these protocols are insecure or prone to attacks. In [18] Francillon et al. propose tinyRNG - a cryptographic pseudo - random number generator for wireless sensor networks. The generator uses transmission bit errors as a main source of entropy. The authors prove that the bit errors are randomly distributed, uncorrelated and difficult to manipulate. The algorithm contains the entropy accumulator part and the ciphering part. The entropy accumulator is built from CBC-MAC function and when its output has sufficient entropy it is used as the input - seed for Cryptographic Pseudo - Random Number Generator, which is a block cipher in counter mode. The outputs of the block cipher are used as the pseudo-random numbers.

The algorithm requires that the initial key - seed for the block cipher is preloaded at the programming time and stored in the internal memory. There

are as many seeds as many nodes are in the network. After the node has booted the initial seed is used to compute a new seed, which is then saved in memory. The tinyRNG requires the communication between nodes in order to produce the pseudo-random numbers.

The random number generator presented by Lo Re et al. in [33] uses the sensing properties of the wireless sensor network. The main assumption here is that every sensor node in the network is able to perform sensing in order to generate random numbers and these are generated using the ADC converter. The measurements are then buffered and encrypted using CMAC algorithm. The result of the encryption is XOR-ed with a register block what results in random number.

To avoid possible manipulation attacks, the sensing task is performed by a randomly chosen node being the neighbour of the node requesting the random number. Thus, this approach requires additionally that nodes cooperate and communicate to obtain the random numbers.

In [19] Gaglio et al. propose the True Random Number Generator using non-deterministic sensor measurements (temperature, humidity and light exposure sensors). The sensor measurements are used as seeds of the algorithm improving the statistical properties of the generated random sequences by adding the diffusion and confusion and by de-skewing. For this purpose MAC based algorithms are used, i.e. either HMAC or CMAC. The TRNG requires a node acting as a Base Station which is responsible for processing the data from sensors and producing the random number sequences. The approach requires the communication between nodes and the Base Station in order to produce random numbers.

Another pseudo-random number generator for low power wireless sensor networks is proposed by Seetharam et al. in [51]. The generator is based on timer available on the sensor platform and its values are XOR-ed with the key. The initial value of the key is the ID of the node and the key is changed by the ones' complement of the timer value. Additionally it is updated with the 8-bit CRC values of the transmitted and received packets. The timer value is changed by ones' complement of the generated random number.

Another approach for low power devices is proposed by Wang et al. in [58]. It utilizes the single electron phenomena and its architecture consists of two parts: single electron circuit and post-processing circuit. The outputs produced by the generator do not present a random behaviour, thus the additional methods for bits skipping or bit counting are used in order to prevent from correlation and to improve the statistical properties. But even after performing these logic operations on the bit stream, the new output failed the tests for randomness recommended by NIST [43]

Chapter 4

shortECC Cryptosystem

This chapter presents the results of the analysis of the cryptographic applicability of elliptic curves over prime fields of small orders that are considered to be too insecure to be used for the cryptographic purposes. It presents the concept and the overall idea of the proposed shortECC approach together with the definition of set-ups in which such lightweight elliptic curves cryptography could be applied. The chapter concludes with the theoretical analysis of the security of shortECC and also with the comparative evaluation of shortECC and other state of the art security approaches for low power devices. The following section describes the main assumptions and features of the proposed approach and points out the differences between the standard ECC and shortECC.

shortECC - Notation

The following notation is used in order to introduce the shortECC approach:

- $\mathbb{F}_p$ - a finite prime field of order p
- $E(\mathbb{F}_p)$ - an elliptic curve over field $\mathbb{F}_p$ given by the equation:

$$Y^2 = X^3 + aX + b \tag{4.1}$$

  where parameters $a, b \in \mathbb{F}_p$.

- (x, y), where $x, y \in \mathbb{F}_p$ - a point on the elliptic curve E
- $\mathcal{O} \in E(\mathbb{F}_p)$ - a point at infinity
- $P = (x_P, y_P)$ - a base point on the elliptic curve E
- $\#E = n$ - order of the elliptic curve

- n, where $nP = \mathcal{O}$ - order of the point P
- A key pair (k, Q), where the private key $k \in [0, p]$ and the public key is $Q = kP$
- $< P >$ - cyclic group of points generated by P

4.1 shortECC Overview

The idea behind the shortECC cryptography is to use elliptic curves which are defined over finite fields of prime order, where the bit length of the order is between 32 and 64 bits. Such cryptography is foreseen for low power devices and can be applied in WSN only if several requirements are fulfilled and several constraints are followed. They concern both the environment in which shortECC is used and the mathematical properties of the sub-components used to build the cryptosystem. These requirements and constraints are described in following sub-sections.

4.1.1 Public/private domain for the security parameters

Due to the usually insecure key lengths used in shortECC, the cryptosystem is usable only for closed groups of nodes. The standard cryptography based on elliptic curves assumes that only the private key is kept secret and the remaining parameters, i.e.: $\mathbb{F}_p, a, b, P, n, Q$ are publicly known. The main assumption of the shortECC approach is that $\mathbb{F}_p, a, b, P, n, Q$ are known only within the trusted and closed group of nodes. Each trusted node has also a key pair consisting of the public and the secret key. The public keys are known only to the trusted group members. The above mentioned parameters will be referred as the shortECC parameters (in contrast to the standard ECC parameters) in the rest of this thesis. This thesis assumes the existence of mechanisms for authentication of the trusted group members. The changes applied to the parameter set - some public parameters become private - causes shortECC to become a Public/Private Key Cryptography hybrid, but also causes the method to be very efficient, because it enables using much shorter key lengths as in case of standard ECC. The efficiency advantages include reduction of the computational effort, but also reduction of the data block size, and thus reduction of energy needed for the transmitting of the results. The comparison of the shortECC and other standard approaches in terms of computational effort and ciphertext size is presented further in this chapter.

4.1.2 Access authority in trusted group

One of the nodes within the trusted group, e.g., a base station, having usually more energy and computational resources than the rest of the nodes, performs the role of the access authority. This node manages the group memberships and is a kind of gateway between the trusted group and the rest of the world. A node that wants to join the trusted group has to be considered trustworthy by the access authority. In order to prove that, it has to have a ticket that grants the access to the group. The ticket has to be issued and signed by some third party, which is assumed to be trustworthy by the access authority. Issuing the tickets and the role of the third party are out of scope in this thesis and these functionalities are assumed to exist. The access authority is responsible for secure distribution of the shortECC parameters among the sensor nodes in the group and for the authentication of the group members, if necessary.

4.1.3 Changing the shortECC parameters

Changing the shortECC parameters periodically in order to enhance the trusted group security and thus making the brute force attack harder. However, is not necessary in case when there is no suspicion of any kind of attack in the group. Results of experiments, as presented in section 4.3, show that the time needed to reveal the shortECC parameters, when applying the exhaustive search method is counted in billions of years. The shortECC parameters should be changed whenever it is detected that a group member was compromised. The generation of the shortECC parameters is not deterministic - it requires choosing a new set of parameters, i.e., prime field or elliptic curve equation parameters, until they fulfil the shortECC requirements. Thus, the method can be computationally too expensive for the low power devices and the generation procedure should be performed by one more powerful node, e.g. the one being also the access authority in the closed group and being able to distribute the new shortECC parameters securely, e.g. using the parameters' management tool.

4.1.4 Distribution of shortECC parameters in the trusted group

It is assumed that all the nodes in the trusted group are equipped with the standard ECC parameters and each node has its own public/private key pair. The standard ECC is used only in special cases that are described further in this thesis and uses the same algorithms and operations as the shortECC cryptosystem. The closed group of nodes willing to use the shortECC cryptosystem requires a parameters' management tool responsible for secure distribution of the shortECC parameters. There are the following cases that need to be considered by that tool:

- The group is not affected by any anomalous situation or behaviour inside the group and also from outside of the group - in this case the parame-

ters' management tool uses currently used shortECC in order to securely distribute and exchange the new shortECC parameters between the sensor nodes in the group.

- An anomaly has been detected - In this case the parameters' management tool uses the standard ECC for the secure distribution of new shortECC parameters.

- The currently used shortECC parameters are provided to the new and authenticated group members using the standard ECC.

4.1.5 Application area

The shortECC is especially usable to secure short data payloads, i.e. those with a bit length not exceeding the shortECC parameters bit lengths. Such cases include most of the sensor measurements, which are often short data payloads and usually require less than 32 bits. An attempt of securing data with length greater than the shortECC parameters, requires splitting the data in blocks of that size, what provides multiple shortECC compliant packets, each requiring separate shortECC encryption operations.

4.1.6 Capabilities of an adversary

Within the framework of this thesis it is assumed that the only type of adversary being interested in data secured with shortECC cryptosystem is a combination of mote-class attacker and a laptop-class attacker. Such attacker can either compromise low-power sensor nodes or perform attacks (brute force) using devices with stronger computational and energy resources than the sensor nodes are equipped with.

4.1.7 Point compression

A very important feature of shortECC is that the points are always sent in the compressed form (see Section 2.2.1). This approach known from the standard ECC gets an additional security meaning in case of the shortECC cryptosystem. Instead of sending both coordinates, only the x-coordinate is sent together with the information about the sign of the y-coordinate. Only the group members knowing the shortECC parameters are able to determine the second coordinate. Compression reduces the amount of data to be transmitted i.e. it reduces the length of the ciphertext and additionally improves the security (Section 4.3.3). Since shortECC parameters are available only to the group members, an eavesdropper can only try to guess the y-coordinate, e.g., by applying a brute force attack or performing the attack described in Section 4.3.4. Finding the proper

shortECC parameters on the basis of an eavesdropped coordinate of some unknown point is by far more complicated than doing the same knowing both coordinates. Sending only the x-coordinate causes that the total amount of data to be sent is 66-bit in case of an encryption and 98-bit in case of a digital signature for input data of 32-bit length, what reduces the transmission energy costs.

4.1.8 Selecting the prime fields

The shortECC requires selecting the prime fields fulfilling the following condition: Let $\mathbb{F}_p$ be a prime field, such that:

$$p \equiv 3 \pmod 4 \tag{4.2}$$

In this case the method for finding the square root modulo p for a given X, i.e. solving the elliptic curve equation:

$$Y^2 = X^3 + aX + b \tag{4.3}$$

reduces to computing $(X^3 + aX + b)^{(p+1)/4}$ [25] and is less complicated, compared to the Tonelli-Shanks algorithm [8] involving randomly choosing elements fulfilling some defined conditions and using the power operation several times.

4.1.9 Non-singular elliptic curves

For the elliptic curves $E(\mathbb{F}_p)$ represented by the equation

$$Y^2 = X^3 + aX + b \tag{4.4}$$

it should be assured that the discriminant $4a^3 + 27b^2$ of the elliptic curve fulfils the following condition:

$$4a^3 + 27b^2 \neq 0. \tag{4.5}$$

4.1.10 Non-anomalous elliptic curves

The elliptic curve $E(\mathbb{F}_p)$ is anomalous if

$$\#E(\mathbb{F}_p) = p. \tag{4.6}$$

For this type of elliptic curves the ECDLP can be efficiently solved [21] and thus they are not recommended to be used for cryptographic purposes.

4.1.11 Elliptic curves of prime order

Only elliptic curves with a prime number of points can be used in shortECC. It improves the overall security of the shortECC based cryptosystem, preventing

from Pohlig-Helmann attack and causing the brute force attack to be more complicated, (see Section 4.3). For these curves all the points on the curve constitute a single cyclic group. Thus, any point in this group is a generator of all the other points, i.e., any point on the curve can be used as the base point.
Let n be the prime order of $E(\mathbb{F}_p)$ and $Q = kP_i$, where $k \in \mathbb{F}_p$, $P_i \in E$ and $i \in [1, n-1]$, then the group of points on E looks as follows:

$$E = \{\mathcal{O}, P_1, P_2, \ldots, P_{n-1}\}. \tag{4.7}$$

The shortECC assumption is that the group of points on the elliptic curve is cyclic and it has prime order. There are no subgroups of E and there are no points ($\neq \mathcal{O}$) with an order different from n. Thus, any point from this group can be a generator of E. So there are $(n-1)$ possible generators of E:

$$E = \{\mathcal{O}, P_1, 2P_1, \ldots, (n-1)P_1\} \tag{4.8}$$
$$E = \{\mathcal{O}, P_2, 2P_2, \ldots, (n-1)P_2\} \tag{4.9}$$
$$\vdots \tag{4.10}$$
$$E = \{\mathcal{O}, P_{(n-1)}, 2P_{(n-1)}, \ldots, (n-1)P_{(n-1)}\} \tag{4.11}$$
$$\tag{4.12}$$

Where $< P_1 >=< P_2 >=< P_3 >= \cdots =< P_{(n-1)} >$ and

$$\forall_{i,j,k \in [1,n-1]}\ iP_j \neq iP_k. \tag{4.13}$$

It can be seen that the point Q can be obtained in each of the $(n-1)$ above mentioned permutations:

$$\forall_{i \in [1,n-1]} \exists_{k \in [1,n-1]}\ kP_i = Q. \tag{4.14}$$

Additionally, to prevent from Pohlig-Hellman attack [4] it is necessary that the order of the base point is the largest possible prime integer. The ideal solution is to find a point which generates all the points on the curve E and has a prime order, equal to the curve order. Lagrange's theorem says [4] that the order of an arbitrarily chosen group element $P \neq \mathcal{O}$ divides the order of the group. Thus, when a curve has a prime order n, there are only two possible orders of points on the curve: 1 and n, and since $1P = P \neq \mathcal{O}$, the only possible order of any point P is n. The Hasse's theorem [26] brings very useful information about the interval for the possible curve order and in the above mentioned case it is also the order of the base point. This theorem says that for an elliptic curve E over a finite field $\mathbb{F}_p$ the order of $E(\mathbb{F}_p)$ satisfies

$$p + 1 - 2\sqrt{p} \leq \#E(\mathbb{F}_p) \leq p + 1 + 2\sqrt{p} \tag{4.15}$$

The following procedure finds the curve order in this interval:
Let $P = (x, y)$ be a randomly chosen point on E and let

$$Q = (p + 1 + 2\sqrt{p})P \tag{4.16}$$

The next step is to find a second point $R = zP$, where $z \neq (p + 1 + 2\sqrt{p})$ and $R = Q$. Since $Q - R = \mathcal{O}$ then

$$t = p + 1 + 2\sqrt{p} - z \tag{4.17}$$

is the potential order of the point P.
The random Rho Pollard method computing collisions of elements of a finite group [4] can be used to find z. The collision is found after approximately $\sqrt{p\pi/2}$ iterations. If the collision occurs, the last thing to check is if t is prime and if it belongs to the Hasse range. If this is the case, then the base point and its order are found, else the procedure needs to be repeated for new curve parameters (a and b), value p and the prime field can also be changed. For primality testing of small integers (< 341550071728321) a deterministic variant of Miller-Rabin Primality Test can be used [22].

4.1.12 Selecting the sub-components for shortECC

The shortECC is designed to be used in resource contained environments. The short parameters' lengths should cause less computational effort than standard ECC. Thus, also the selection of the sub-components used in shortECC should be done carefully. The sub-components are the mechanisms responsible for the following functionalities:

- generation of the cryptographic secure pseudo-random numbers used for instance as secret keys or as parameters in encryption operation;
- hash function used in digital signature algorithms;
- algorithm used for embedding the plain text message into a point on the elliptic curve.

The selection of the mechanisms for shortECC cryptosystem and the motivation behind that are explained in the following paragraphs.

Pseudo-random number generators

The requirements of the shortECC dictate provision of the cryptographic pseudo-random numbers of lengths that are suitable for computations in shortECC, i.e. lengths not exceeding the lengths of shortECC parameters. The state of the art cryptographic pseudo-random number generators [19] or [18] operate on numbers

that are significantly larger than shortECC parameters and often do not fulfil the requirements for cryptographic randomness. For the needs of shortECC a new approach called lmRNG [54] was proposed and it is described in Chapter 5. It is computationally inexpensive and consuming minimal energy, compared to the state of the art solutions and in contrast to these it does not involve any hardware besides the one already available on the test platform. Additionally, it produces cryptographic secure pseudo-random numbers, what was tested (see Section 5.3.3).

Hash functions

Standard digital signature schemes [26] use cryptographic hash functions that produce outputs of 160 or even 512 bits, thus they are not applicable in the shortECC authentication protocol. Even the lightweight proposals offer outputs that are too long for the shortECC approach, e.g. the H-Present hash functions offering outputs between 64 and 128 bits [48]. Additional drawback of the state of the art hash functions is that they involve block ciphers or other ciphering methods for generating the hash messages, and thus, require effort related to the implementation of additional algorithms and extra memory space on the test platform. The shortECC proposes an authentication scheme, but without using any hash function. This scheme is described further in Section 4.2.2.

Embedding the plain-text message into point

In order to encrypt the message using the elliptic curve cryptography, it is necessary to embed the message into a point lying on the elliptic curve. For the prime fields chosen for the shortECC cryptosystem, i.e., those with the property:

$$p \equiv 3 \pmod 4 \tag{4.18}$$

where, p is the order of the prime field, the method for embedding the message into a point is as follows [25]: Let m be some plain text message such that:

$$0 \leq m \leq p/1000 - 1 \tag{4.19}$$

The three least significant digits (decimal) are append to m until there is an x, where

$$1000m \leq x \leq 1000(m+1) \leq p \tag{4.20}$$

such that

$$f(x) = x^3 + ax + b \tag{4.21}$$

is a square in $\mathbb{F}_p$. Then $P = (x, f(x)^{(p+1)/4})$ is the point on the elliptic curve $E(\mathbb{F}_p)$ that represents the message m. And the message m can be decoded from the point P by dropping the three least significant digits (decimal) from the x-coordinate.

4.2 Protocols in shortECC

The shortECC approach provides two possibilities for securing the data: encryption (and decryption) and the digital signature/authenticated encryption. The methods are described in following two sections.

4.2.1 Encryption and Decryption

In shortECC the encryption and decryption operations are performed using the Elliptic Curves ElGamal algorithm (see Algorithm 8). The algorithms are suitable also for standard ECC and since they do not involve any additional subcomponents, e.g. hash function, they can be used in the unchanged form also in shortECC. The algorithm used for encryption requires secure pseudo-random number generation, what in case of shortECC is ensured by the lmRNG generator (see Chapter 5). The appropriate generation of the pseudo-random numbers is crucial to the overall security of the ElGamal encryption [57].

4.2.2 Digital Signature/Authenticated Encryption with message recovery

The standard ECDSA protocol for digital signature requires using a cryptographic hash function, see Algorithm 9. As mentioned previously, due to the large input and output numbers, such functions are inapplicable in the shortECC environment. Thus, a modified version of standard Elliptic Curves Digital Signature Algorithm is proposed within the framework of this thesis, for the digital signature. It is suitable for both shortECC and the standard ECC cryptography and its security is discussed later in this section. The signature algorithm is a 1-to-1 signature without appendix, but with message recovery. The message is signed with the signer's private key and encrypted with the verifier's public key. The signing phase results in an integer - the signature and two points - the encrypted message. The signature is computed using the signer's private key, two coordinates of the point representing the message and a pseudo-random integer. This pseudo-random integer is also used for encryption of the message. In the verification phase, the verifier uses its private key to decrypt the message and the signer's public key to verify the signature. The verification is possible only in case when the decryption of the message is correct, since its coordinates are needed in order to verify the signature. The advantage of this method, compared to symmetric solutions, like Message Authentication Code and Encryption, is that there is no need to have the common keys for each pair of the group members. The authenticity of the message sent by the sender can be verified by all the nodes in the trusted group knowing the sender's public key.

The signing procedure is presented in Algorithm 11, while the verification procedure is presented in Algorithm 12.

Algorithm 11 shortECC Digital Signature with message recovery - signing

Summary: There is a set of shortECC parameters, base point P on the elliptic curve, private key k_S of the signer, public key Q_V of the verifier, message m.

Result: The signature (R, S, s).

The signer performs the following steps:

Converts the message m into a point $M = (x_M, y_M)$ on the shortECC elliptic curve.

Generates a pseudo-random integer $k \in (0, p-1)$.

Computes $s = k^{-1}(x_M + y_M k_S) \pmod{p}$.

Encrypts message point M using the same pseudo-random k and the public key Q_V of the verifier:

$R = kP$

$S = M + kQ_V$

The signature is a triple (R, S, s).

Algorithm 12 shortECC Digital Signature with message recovery - verification

Summary: There is a set of shortECC parameters, base point P on the elliptic curve, private key k_Q of the verifier, public key Q_s of the signer, signature (R, S, s).

Result: Acceptance or rejection of the signature, the message m.

The verifier performs the following steps:

Decrypts R and S to obtain the coordinates (x_M, y_M) of M needed in the next steps of verification:

$M = S - k_Q R$.

Computes two values u_1, u_2:

$u_1 = s^{-1}x_M, \qquad u_2 = s^{-1}y_M$.

Verifies the signature by comparing R with:

$u_1 P + u_2 Q_S$

If the comparison is true then return ACCEPT SIGNATURE and convert M into m; else return REJECT SIGNATURE.

The correctness of the verification procedure of the signature (R, S, s) can be proved in following way:

$$\begin{aligned} u_1 P + u_2 Q_S &= s^{-1}x_M P + s^{-1}y_M Q_S && (4.22)\\ &= k(x_M + y_M k_S)^{-1} x_M P + && (4.23)\\ &+ k(x_M + y_M k_S)^{-1} y_M k_S P && (4.24)\\ &= k[(x_M + y_M k_S)^{-1}(x_M + y_M k_S)]P && (4.25)\\ &= kP = R && (4.26) \end{aligned}$$

Security of the proposed digital signature

The proposed approach is applicable also in the standard ECC environment under the assumption that the ECDLP problem is hard to solve. The signature consists

of three elements: two points and an integer. Trying to get any information, i.e. private key of the verifier, from these two points is possible only if the ECDLP can be solved. The third element contains the private key of the signer and in order to prevent this private key from being disclosed following assumptions need to be taken into consideration:

- The random element k needs to be changed for each new signature, otherwise the private key is unsafe and can be revealed, (see Note 4.35 in [21]).

- The coordinates of the message are also a part of the integer element of the signature and knowing them it would be easier to discover the private key of the signer. Thus, the message has to be sent to the verifier in the encrypted form and only the verifier is able to decrypt it first and then to verify the signature. Access to these coordinates is possible either after decrypting the message from the two points available in the signature or after solving the ECDLP.

In case of shortECC the security of the proposed protocol is guaranteed by keeping the shortECC parameters secret.

4.2.3 shortECC key pair generation

At the start-up each node in the trusted group is equipped with its private/public key pair and is able to confirm its identity to the access authority by presenting the ticket granted by the trusted third party. The cryptosystem based on shortECC uses a cryptographic secure pseudo-random number generator, which can be used by the members of the trusted group to generate the private keys. Public keys are generated by multiplying the currently used base point by the generated private keys. These are then distributed among the sensor nodes in the group using a parameter management tool, e.g. using the standard ECC.

4.3 Security Analysis

The following section considers the attempts of an adversary to break the shortECC cryptosystem. Compared to the standard ECC the task of an attacker is slightly different, i.e. it is not sufficient to just obtain the secret key, but she also has to find the right elliptic curve parameters.

The computations and estimations were performed for 32-bit long shortECC parameters. The first scenario concerns finding the shortECC parameters on the basis of the eavesdropped x-coordinate of some unknown point. The second scenario considers the ambiguity of the results even if the shortECC parameters are known.

4.3.1 Determining the shortECC parameters on the basis of the known x-coordinate

Let us assume that an adversary has an eavesdropped x-coordinate, which is 32-bit long. In order to determine the proper shortECC parameters she has to separate the set of elliptic curves on which a point with this x-coordinate exists and to do this she needs all the elliptic curves over all 32-bit long prime fields.

In the first step it is necessary to find all the 32-bit long prime numbers, i.e. all orders q of primes field $\mathbb{F}_q$, which fulfil the condition that $(q \mod 4 = 3)$. According to [38] the number of prime numbers less than z can be approximated as follows:

$$\frac{z}{\ln z}. \tag{4.27}$$

In order to estimate the number of 32-bit long prime numbers it is necessary to determine the smallest and the largest 32-bit primes serving as boundaries of the range containing all the 32-bit long primes. The two numbers were generated using simple Java application and verified on the web site [10]. The range determined by the smallest and largest 32-bit prime is taken as a search area and only the finite fields lying in this range will be considered as 32-bit long (prime) fields in the next computations. The smallest 32-bit prime is

$$2\,147\,483\,659 \tag{4.28}$$

and the largest 32-bit prime is

$$4\,294\,967\,291. \tag{4.29}$$

Then, the number of 32-bit long primes can be approximated as follows:

$$\frac{4\,294\,967\,291}{\ln 4\,294\,967\,291} - \frac{2\,147\,483\,659}{\ln 2\,147\,483\,659} = 193\,635\,250 - 99\,940\,775 = 93\,694\,475 \tag{4.30}$$

Since all prime numbers are odd numbers, computing $(q \mod 4)$ gives as a result either 1 or 3. Thus, the number of primes that are relevant for shortECC approach is assumed to be about:

$$\frac{93\,694\,475}{2} = 46\,847\,236 \tag{4.31}$$

The number of possible elliptic curves over one prime finite field $\mathbb{F}_q$ equals to $q^2 - q$ [30]. Hence, the number of elliptic curves over the prime field which order is the smallest 32-bit prime is equal to:

$$2\,147\,483\,659^2 - 2\,147\,483\,659 = 4\,611\,686\,063\,524\,544\,622 \tag{4.32}$$

And, the number of elliptic curves over the prime field which order is the greatest 32-bit prime is equal to:

$$4\,294\,967\,291^2 - 4\,294\,967\,291 = 18\,446\,744\,035\,054\,845\,972 \tag{4.33}$$

Thus, the average number of elliptic curves over an arbitrary finite field which order is 32-bit prime can be estimated as follows:

$$\frac{4\,611\,686\,063\,524\,544\,622 + 18\,446\,744\,035\,054\,845\,972}{2} = 11\,529\,215\,049\,289\,695\,297 \tag{4.34}$$

Hence, for all 32-bit prime fields relevant for shortECC there are approximately

$$11\,529\,215\,049\,289\,695\,297 * 46\,847\,236 = 540\,111\,858\,308\,825\,987\,946\,649\,092 \tag{4.35}$$

potential elliptic curves.

The shortECC assumption is that the group of points on the elliptic curve has a prime order. In order to determine if the elliptic curve has prime order it is necessary to count the points on the curve and check the primality of the number of points. Generating and checking the primality of one such an elliptic curve takes about 6 seconds on a computer with Intel Core2 Duo P9600 processor and 4Gbytes of RAM.

The number of the elliptic curves of prime order can be estimated as follows: According to the Hasse's Theorem [21] it is possible to compute the interval (called Hasse interval) to which the order of the elliptic curve belongs. For the elliptic curve E over prime field $\mathbb{F}_q$ the order of the elliptic curve is in the interval:

$$(q + 1 - 2\sqrt{q}, q + 1 + 2\sqrt{q}). \tag{4.36}$$

For the above mentioned smallest 32-bit long prime number, i.e. for the $q = 2\,147\,483\,659$ the Hasse interval is:

$$(2\,147\,390\,978, 2\,147\,576\,341). \tag{4.37}$$

Thus, there are 185 363 possible orders of the elliptic curve in this interval, and the number of possible prime orders in this interval can be estimated as follows:

$$\frac{2\,147\,576\,341}{\ln 2\,147\,576\,341} - \frac{2\,147\,390\,978}{\ln 2\,147\,390\,978} = 8225. \tag{4.38}$$

For the above mentioned greatest 32-bit long prime number, i.e. for the $q = 4\,294\,967\,292$ the Hasse interval is:

$$(4\,294\,836\,221, 4\,295\,098\,365). \tag{4.39}$$

Thus, there are 262144 possible orders of the elliptic curve in this interval, and the number of possible prime orders in this interval can be estimated as follows:

$$\frac{4\,295\,098\,365}{\ln 4\,295\,098\,365} - \frac{4\,294\,836\,221}{\ln 4\,294\,836\,221} = 11\,286. \tag{4.40}$$

The average number of possible orders for the arbitrary 32-bit field $\mathbb{F}_q$ can be estimated as follows:

$$\frac{185\,363 + 262\,144}{2} = 223\,754. \tag{4.41}$$

Thus for all 32-bit prime fields there are approximately:

$$93\,694\,475 * 223\,754 = 20\,964\,513\,559\,150 \tag{4.42}$$

possible orders of the elliptic curves.

Hence, the number of elliptic curves per order can be estimated as follows: Assuming an equal distribution of the elliptic curves over the possible orders there are approximately:

$$\frac{540\,111\,858\,308\,825\,987\,946\,649\,092}{20\,964\,513\,559\,150} = 25\,763\,147\,653\,531 \tag{4.43}$$

elliptic curves of an 32-bit order per finite prime field.

The average number of possible prime orders for the arbitrary field $\mathbb{F}_q$ can be estimated as follows:

$$\frac{8225 + 11\,286}{2} = 9756. \tag{4.44}$$

Thus, the number of elliptic curves of an 32-bit prime order over 32-bit prime fields is as follows:

$$9756 * 2\,576\,314\,765\,353 = 25\,134\,526\,850\,783\,868. \tag{4.45}$$

In order to select the elliptic curves on which a point with x-coordinate exists, the adversary has to solve approximately 25 134 526 850 783 868 elliptic curve equations. Finding one square root modulo, i.e., solving the elliptic curve equation takes about 100 microseconds on a computer with Intel Core2 Duo P9600 processor and 4Gbytes of RAM. For all the curves it would take about 79 646 509 years. This task can easily be split in almost any number of parallel tasks, but even assuming an ideal parallelization overhead and having 80 000 000 above mentioned processors available, it would take about a year to solve it.

4.3.2 Ambiguity of the results for known shortECC parameters

Let x_Q be the eavesdropped x-coordinate of some unknown point. Let us assume that the adversary has a set of elliptic curves, each of them having the point with x_Q coordinate. Let $E(\mathbb{F}_q)$ be one of the curves from the set and $Q \in E$ be the point with x_Q coordinate. The x_Q coordinate belongs to a point which is a multiple of the base point P chosen for the cryptosystem, i.e. $Q = (x_Q, y_Q) = kP$, where the integer k is unknown. If the adversary is able to determine the base point on some elliptic curve, then it is possible to break the security system. As mentioned in Section 4.1.11, the points on elliptic curves chosen for shortECC cryptosystem are always cyclic groups of prime order. If the order of $E(\mathbb{F}_q)$ is a prime number n, then there are $n - 1$ possible cyclic groups of n points on this elliptic curve, where each cyclic group is created by a different point on E. Thus, every cyclic group consists of all points on the elliptic curve and in every cyclic

group the point with x_Q coordinate can be found. But, in every cyclic group the point with x_Q coordinate is created by multiplying a different base point by a different integer. The following formulas present the cyclic groups of points on E each having different base point, i.e. $P_1, P_2, P_3, \ldots, P_{(n-1)}$ respectively.

$$E = \{\mathcal{O}, P_1, 2P_1 = (x_Q, y_Q), \ldots, (n-1)P_1\} \tag{4.46}$$
$$E = \{\mathcal{O}, P_2, 2P_2, \ldots, 123P_2 = (x_Q, y_Q), \ldots, (n-1)P_2\} \tag{4.47}$$
$$E = \{\mathcal{O}, P_3, 2P_3, \ldots, 44P_3 = (x_Q, y_Q), \ldots, (n-1)P_3\} \tag{4.48}$$
$$\vdots \tag{4.49}$$
$$E = \{\mathcal{O}, P_{(n-1)}, 2P_{(n-1)}, \ldots, (n-1)P_{(n-1)} = (x_Q, y_Q)\} \tag{4.50}$$

It can be seen that in each cyclic group the point with x_Q coordinate is created from a different base point. Thus, if the adversary has a suspected elliptic curve with a point having x_Q as a coordinate, the probability of finding the proper base point is equal to

$$\frac{1}{n-1} \tag{4.51}$$

4.3.3 Attack on shortECC: finding the modulus and equation parameters on the basis of points in an uncompressed form

Let $E(\mathbb{F}_p)$ be the shortECC elliptic curve represented by equation

$$Y^2 = X^3 + aX + b. \tag{4.52}$$

Let $S = (x_S, y_S), T = (x_T, y_T), R = (x_R, y_R)$ be the eavesdropped points and both the coordinates of these points are known. In order to determine the p, a, b parameters an adversary forms the following equations:

$$\begin{cases} y_S^2 = x_S^3 + ax_S + b \pmod p \\ y_T^2 = x_T^3 + ax_T + b \pmod p \\ y_R^2 = x_R^3 + ax_R + b \pmod p \end{cases} \tag{4.53}$$

And after that, she computes:

$$\begin{cases} y_S^2 - y_T^2 = x_S^3 + ax_S + b - (x_T^3 + ax_T + b) \pmod p \\ y_T^2 - y_R^2 = x_T^3 + ax_T + b - (x_R^3 + ax_R + b) \pmod p \\ y_S^2 - y_R^2 = x_S^3 + ax_S + b - (x_R^3 + ax_R + b) \pmod p \end{cases} \tag{4.54}$$

What after reducing gives:

$$\begin{cases} y_S^2 - y_T^2 - x_S^3 + x_T^3 = a(x_S - x_T) \pmod p \\ y_T^2 - y_R^2 - x_T^3 + x_R^3 = a(x_T - x_R) \pmod p \\ y_S^2 - y_R^2 - x_S^3 + x_R^3 = a(x_S - x_R) \pmod p \end{cases} \tag{4.55}$$

Since $x_S, y_S, x_T, y_T, x_R, y_R$ are all known, the equations reduce to:

$$\begin{cases} n = ak \pmod p \\ n' = ak' \pmod p \\ n'' = ak'' \pmod p \end{cases} \tag{4.56}$$

with unkown a, p parameters. What leads to:

$$\begin{cases} nak' = akn' \pmod p \\ n'ak'' = ak'n'' \pmod p \\ n''ak = nak'' \pmod p \end{cases} \tag{4.57}$$

And finally to:

$$\begin{cases} nk' - kn' = 0 \pmod p \\ n'k'' - k'n'' = 0 \pmod p \\ n''k - nk'' = 0 \pmod p \end{cases} \tag{4.58}$$

The last thing to compute is the greatest common prime divisor of the three above left-hand values. The number can be considered as the probable prime modulus p.

The following example illustrates the above considerations. Let $E(\mathbb{F}_7)$ be the secret shortECC elliptic curve represented by the equation

$$Y^2 = X^3 + 2X + 4 \pmod 7 \tag{4.59}$$

Let us assume that an adversary has four eavesdropped points $S = (2,3), T = (3,4), R = (6,1), U = (3,3)$ and has no information about the parameters (a, b, p).

Then for the three points $S = (2,3), T = (3,4), R = (6,1)$ she can compute:

$$\begin{cases} 3^2 = 2^3 + 2a + b \pmod p \\ 4^2 = 3^3 + 3a + b \pmod p \\ 1^2 = 6^3 + 6a + b \pmod p \end{cases} \tag{4.60}$$

And after that, she computes:

$$\begin{cases} 3^2 - 4^2 - 2^3 + 3^3 = a(2-3) \pmod p \\ 4^2 - 1^2 - 3^3 + 6^3 = a(3-6) \pmod p \\ 3^2 - 1^2 - 2^3 + 6^3 = a(2-6) \pmod p \end{cases} \tag{4.61}$$

The equations boil down to:

$$\begin{cases} 12 = a(-1) \pmod p \\ 204 = a(-3) \pmod p \\ 216 = a(-4) \pmod p \end{cases} \tag{4.62}$$

What leads to:

$$\begin{cases} 12a(-3) = 204a(-1) \pmod p \\ 12a(-4) = 216a(-1) \pmod p \\ 204a(-4) = 216a(-3) \pmod p \end{cases} \tag{4.63}$$

And finally to:

$$\begin{cases} 168 = 0 \pmod p \\ 168 = 0 \pmod p \\ -168 = 0 \pmod p \end{cases} \tag{4.64}$$

And for the three points $S = (2, 3), T = (3, 3), R = (6, 1)$ she can compute:

$$\begin{cases} 3^2 = 2^3 + 2a + b \pmod p \\ 3^2 = 3^3 + 3a + b \pmod p \\ 1^2 = 6^3 + 6a + b \pmod p \end{cases} \tag{4.65}$$

What gives:

$$\begin{cases} 19 = a(-1) \pmod p \\ 197 = a(-3) \pmod p \\ 216 = a(-4) \pmod p \end{cases} \tag{4.66}$$

What leads to:

$$\begin{cases} 19a(-3) = 197a(-1) \pmod p \\ 19a(-4) = 216a(-1) \pmod p \\ 197a(-4) = 216a(-3) \pmod p \end{cases} \tag{4.67}$$

And finally to:

$$\begin{cases} 140 = 0 \pmod p \\ 140 = 0 \pmod p \\ -140 = 0 \pmod p \end{cases} \tag{4.68}$$

Taking the results from above computations one can compute the greatest common prime divisor for 140 and 168:

$$140 = 2 * 2 * 5 * 7 \tag{4.69}$$

And

$$168 = 2 * 2 * 2 * 3 * 7 \tag{4.70}$$

The number determined and suspected to be the prime field order is 7 what is actually the order of the prime field in this example.

4.3.4 Attack on shortECC: finding the modulus and equation parameters on the basis of points in compressed form

Let $E(\mathbb{F}_p)$ be the shortECC elliptic curve represented by equation

$$Y^2 = X^3 + aX + b. \tag{4.71}$$

Let $x_1, x_2, \ldots, x_n$ be the eavesdropped x-coordinates of the points on the elliptic curve. In order to determine the y-coordinates $y_1^2, y_2^2, \ldots, y_n^2$ and the p, a, b parameters an adversary needs to solve the following system of equations:

$$\begin{cases} y_1^2 = x_1^3 + ax_1 + b \pmod p \\ y_2^2 = x_2^3 + ax_2 + b \pmod p \\ \vdots \\ y_n^2 = x_n^3 + ax_n + b \pmod p \end{cases} \tag{4.72}$$

The above system of equations can be reduced to the system of linear equations by substituting $y_1^2, y_2^2, \ldots, y_n^2$ in the following way:

$$y_1^2 = z_1, y_2^2 = z_2, \ldots, y_n^2 = z_n. \tag{4.73}$$

Hence, the system to be solved looks as follows:

$$\begin{cases} z_1 = x_1^3 + ax_1 + b \pmod p \\ z_2 = x_2^3 + ax_2 + b \pmod p \\ \vdots \\ z_n = x_n^3 + ax_n + b \pmod p \end{cases} \tag{4.74}$$

Sorting the unknowns and knowns gives the following system:

$$\begin{cases} -x_1^3 = ax_1 + b - z_1 + 0z_2 + \cdots + 0z_n \pmod p \\ -x_2^3 = ax_2 + b + 0z_1 - z_2 + \cdots + 0z_n \pmod p \\ \vdots \\ -x_n^3 = ax_n + b + 0z_1 + 0z_2 + \cdots - z_n \pmod p \end{cases} \tag{4.75}$$

which can be further written as a vector equation:

$$\begin{bmatrix} -x_1^3 \\ -x_2^3 \\ \vdots \\ -x_n^3 \end{bmatrix} = a \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + b \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} + z_1 \begin{bmatrix} -1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + z_2 \begin{bmatrix} 0 \\ -1 \\ \vdots \\ 0 \end{bmatrix} + \cdots + z_n \begin{bmatrix} 0 \\ 0 \\ \vdots \\ -1 \end{bmatrix} \tag{4.76}$$

The vector equation is equivalent to a matrix equation in the form:

$$Mu = w \tag{4.77}$$

where

$$M = \begin{bmatrix} x_1 & 1 & -1 & 0 & \dots & 0 \\ x_2 & 1 & 0 & -1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_n & 1 & 0 & 0 & \dots & -1 \end{bmatrix}, u = \begin{bmatrix} a \\ b \\ z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix}, w = \begin{bmatrix} -x_1^3 \\ -x_2^3 \\ \vdots \\ -x_n^3 \end{bmatrix} \tag{4.78}$$

Matrix M is called the coefficient matrix and the matrix Mw of the following form:

$$Mw = \begin{bmatrix} x_1 & 1 & -1 & 0 & \dots & 0 & -x_1^3 \\ x_2 & 1 & 0 & -1 & \dots & 0 & -x_2^3 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_n & 1 & 0 & 0 & \dots & -1 & -x_n^3 \end{bmatrix} \tag{4.79}$$

is named augmented matrix.

On the basis of the above matrices it is possible to determine the number of solutions of the given system of equations. It is done using the ranks of the matrices and their properties which are described in the Rouche-Cappelli theorem, where rank of the matrix is the maximal number of linearly independent (there are no vectors in this matrix which can be written as an combination of the other vectors from this matrix) vectors building the columns (or rows) of the matrix. Thus, the rank of the matrix is always less than or equal to the number of columns and less than or equal to the number of rows in the matrix. The Rouche-Cappelli theorem says that:

Theorem 3. *There is solution of the system of the linear equations if the rank of the coefficient matrix M is equal to the rank of the augmented matrix Mw. Additionally, if the rank of the M and Mw is equal to the number of unknowns (given in matrix u), then there is only one unique solution of the system. Otherwise, the system of equation is undetermined.*

Since the number of columns in the matrix M is greater than its number of rows, the rank of the matrix M is dictated by the number of its rows and it can be at most n. Similarly, the rank of the augmented matrix Mw is dictated by the number of its rows and it can also be at most n. Taking these maximal values one can see that:

$$rank(M) = rank(Mwb) = n \leq n + 2 \tag{4.80}$$

where $n + 2$ is the number of unknowns it this system. Thus, the system undetermined and it is impossible to find the shortECC parameters only having the x-coordinates.

The following example illustrates the above considerations. Let $E(\mathbb{F}_7)$ be the secret shortECC elliptic curve represented by the equation

$$Y^2 = X^3 + 2X + 4 \pmod 7 \tag{4.81}$$

Let $S = (2,3), T = (3,4), R = (6,1), U = (3,3)$ be points on this elliptic curve. Let us assume that an adversary has eavesdropped the x-coordinates of these four points resulting in the following knowledge $S = (2, y_S), T = (3, y_T), R = (6, y_R), U = (3, y_U)$ and no information about the parameters (a, b, p) and also about the y-coordinates of the four points. Hence, the system of equations has the following form:

$$\begin{cases} y_1^2 = 2^3 + 2a + b \pmod p \\ y_2^2 = 3^3 + 3a + b \pmod p \\ y_3^2 = 6^3 + 6a + b \pmod p \\ y_4^2 = 3^3 + 3a + b \pmod p \end{cases} \tag{4.82}$$

The above system of equations is reduced to a system of linear equations by substituting the y-coordinates in the following way:

$$y_1^2 = z_1, y_2^2 = z_2, y_3^2 = z_3, y_4^2 = z_4. \tag{4.83}$$

After sorting the unknowns and knowns the system to be solved has the following form:

$$\begin{cases} -2^3 = 2a + b - z_1 \pmod p \\ -3^3 = 3a + b - z_2 \pmod p \\ -6^3 = 6a + b - z_3 \pmod p \\ -3^3 = 3a + b - z_4 \pmod p \end{cases} \tag{4.84}$$

What results in the following matrix representation:

$$M = \begin{bmatrix} 2 & 1 & -1 & 0 & 0 & 0 \\ 3 & 1 & 0 & -1 & 0 & 0 \\ 6 & 1 & 0 & 0 & -1 & 0 \\ 3 & 1 & 0 & 0 & 0 & -1 \end{bmatrix}, u = \begin{bmatrix} a \\ b \\ z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix}, w = \begin{bmatrix} -2^3 \\ -3^3 \\ -6^3 \\ -3^3 \end{bmatrix} \tag{4.85}$$

The augmented matrix Mw has the following form:

$$Mw = \begin{bmatrix} 2 & 1 & -1 & 0 & 0 & 0 & -2^3 \\ 3 & 1 & 0 & -1 & 0 & 0 & -3^3 \\ 6 & 1 & 0 & 0 & -1 & 0 & -6^3 \\ 3 & 1 & 0 & 0 & 0 & -1 & -3^3 \end{bmatrix} \tag{4.86}$$

Even without computing the exact values of the ranks of the matrices M and Mw and assuming the maximal possible ranks, one can see that:

$$rank(M) = rank(Mw) = 4 \leq 6, \tag{4.87}$$

where 6 is the number of the unknowns in this system. Thus, the system is undetermined what means that on the basis of the eavesdropped x-coordinates it is impossible to find the other unknown parameters.

4.4 Evaluation

This section provides the evaluation of the shortECC, which is compared to different state of the art approaches. The evaluation considers the lifetime, the computational efforts as well as applicability of shortECC.

4.4.1 Elliptic curves versus finite field arithmetic: brute force searching area

The asymmetric key cryptography protocols whose security is based on the hardness of solving DLP or of factorization of large integers, e.g. RSA or ElGamal operate on finite fields and thus the area for searching the security parameters is only the finite field $\mathbb{F}_p = (0, ..., p-1)$. Under the similar circumstances as those assumed for shortECC, i.e. the public parameters are shifted to the private domain of some trusted group, finding these parameters requires determining the suspected orders of the finite fields and after that solving the underlying problem. Thus, reducing the key lengths for these cryptographic protocols causes the exhaustive search attack to be easier as in the case of the security protocols based on reduced elliptic curves parameters. As mentioned in Section 4.3.1, the brute force attack on the shortECC with unknown parameters requires determining the suspected finite fields in the first place and after that for each finite field the appropriate elliptic curves need to be chosen from all the existing ones on the respective finite field. Thus, the search area for the shortECC is similar to a tree, where one parameter value identified as suspected one (for instance the finite field order) generates a large number of branches (elliptic curves) and each of these branches has also large number of own branches (equiprobable base points). Thus, public key cryptography with reduced key lengths has more sense for protocols based on elliptic curves.

4.4.2 shortECC versus DES: lifetime

In shortECC all parameters, like the elliptic curve equation, the base point and its order are known only to the trusted group members. Performing an attack to reveal these secret shortECC parameters can be based on the eavesdropped

Table 4.1: Clock cycles measured on MSP430F5438A as costs of prime fields and elliptic curve operations

	Finite field			
Operation	32-bit	128-bit	160-bit	192-bit
Addition in F_p	272	1091	1272	1433
Multiplication in F_p	306	649 172	1 060 458	1 545 824
Inversion in F_p	4704	1 016 254	1 557 159	2 168 614
Point Addition	12 000	3 215 203	4 697 290	6 378 132
Point Multiplication	409 120	651 662 120	1 259 460 903	2 078 311 103
ElGamal Encryption	920 240	1 309 111 615	2 527 176 991	4 168 103 741
ElGamal Decryption	424 320	658 092 719	1 268 854 497	2 091 047 263

x-coordinates. To check the number of elliptic curves on which a point with the eavesdropped x-coordinate exists, an adversary has to solve 25 134 526 850 783 868 elliptic curve equations, what on the PC with Intel Core2 Duo P9600 processor and 4Gbytes of RAM would take about79 646 509 years, (see Section 4.3.1). In [29] the authors present the lifetime of the Digital Encryption Standard (DES) when using key lengths between 56-bit and 78-bits. The estimations were performed using a Pentium II 450MHz PC that is computationally less efficient than the one used for the tests of shortECC. The lifetime of the 56-bit key is equal to 1110 years, and for the 78-bit key the lifetime is 3 220 000 000 years. It is clear that these key lifetimes for the DES keys would be even shorter, if the used PC would be more performant. Thus, shortECC provides higher security level than DES for comparable key lengths.

4.4.3 shortECC versus standard ECC: computational efforts

In case of shortECC the computational effort needed on the wireless sensor node is reduced, compared to the effort related to the computations needed by the standard ECC. Sensor nodes are usually equipped with 8- or 16-bit microcontrollers, performing computations on words of the length corresponding to their architecture, so the smaller the prime field the less operations need to be performed. The test measurements presented in this section where performed on a MSP430F5438A microcontroller [55] working with a 1MHz clock frequency. The computational costs for the elliptic curves over 128, 160 and 192-bit prime fields F_p recommended by NIST and the elliptic curve over 32-bit prime field are presented in Table 4.1. The Table 4.1 presents the comparison of the clock cycles needed by the MSP430F5438A microcontroller for the basic prime field operations and the elliptic curve operations for the above mentioned curves. Both shortECC and standard ECC use the same C code. The presented results show

Table 4.2: Comparison of shortECC and ECIES

	shortECC	**ECIES**
Secret Parameters Management	standard ECC	standard ECC
Message lengths	max 32-bits	arbitrary
Encryption	shortECC ElGamal	standard ECC, AES, hash
Decryption	shortECC ElGamal	standard ECC, AES

that the shorECC is significantly faster, compared to the standard ECC and requires about one second for encryption and half of a second for decryption. The most important operation in the cryptographic algorithms - point multiplication - is about 1500 times faster for 32-bit shortECC, than for 128-bit standard elliptic curves. Thus, if it is foreseen to secure the group/broadcast communication in a WSN scenario, it may be reasonable to use shortECC in cases when the packets to be secured do not exceed 32 bits.

4.4.4 shortECC versus Elliptic Curves Integrated Encryption Scheme: application scenarios

Since the standard ECC is commonly used for sharing credentials in secret key protocols, e.g. Elliptic Curves Diffie-Helmann (see Algorithm 4), one could try to use the ECC functionalities and algorithms in order to secure the exchanged data. For this purpose the Elliptic Curves Integrated Encryption Scheme (ECIES) can be used (see Algorithm3.2.3). It provides mechanisms for encryption and decryption of data, but besides the ECC algorithms it requires implementation of symmetric encryption functions, e.g. AES and of hash functions. The scheme uses standard ECC in order to be able to work with 128-bit (or larger) outputs of AES and hashing functions. Table 4.2 presents a short comparison of the EC Integrated Encryption Scheme and shortECC implementation. In the considered scenario both shortECC and ECIES use standard ECC for the management of the secret parameters. Taking into consideration the previous comparison of the computational efforts of shortECC and standard ECC it is clear, that in cases when the bit-length of the data to be secured does not exceed 32 it is more energy saving to use shortECC for the encryption/decryption purposes. In this case implementation of the additional security functionalities is not required, because shortECC and standard ECC can use the same algorithms.

Table 4.3: Comparison of shortECC and AES

Security Approach	**shortECC**	**Symmetric Key**
Algorithm	ECC ElGamal	AES
Secret Synchronization	no needed	needed
Unique components	no needed	unique nonce needed
Knowledge about payload length	no needed	needed
Public Key Length (uncompressed)	64 bits	-
Private Key Length	32 bits	128 bits
Private Key Length	32 bits	128 bits
Results for encryption and decryption of 32-bit payload		
Clock Cycles for encryption	920240	19000
Clock Cycles for decryption	423120	22000
Plaintext Length	32 bits	32 bits
Ciphertext Length (compressed)	66 bits	32 bits

4.4.5 shortECC versus AES: encryption and decryption

Shifting the usually public ECC parameters into the private domain makes the shortECC approach comparable to secret key cryptography approaches, like the AES algorithm, which is often used in WSN applications. In contrast to AES the shortECC does not require any additional parameters, like knowledge about the length of the payload or unique nonces. Also the secret synchronization required in AES is not needed in the shortECC approach. Table 4.3 presents the results of comparing the encryption and decryption mechanisms using shortECC and AES. The numbers of clock cycles presented in the table were computed as average values of 1000 encryptions and decryptions performed for each of the methods. The computations were performed on a sensor node equipped with the MSP430F5438A microcontroller.

The performed comparison shows (see Table 4.3) that even the reduced lengths of the parameters in shortECC are not sufficient to make the ECC based cryptography competitive to AES, with respect to the computational effort. The cost of one shortECC encryption can be compared to 48 AES encryptions and the cost of one shortECC decryption is comparable to 19 AES decryptions. However, this does not disqualify the shortECC approach from being useful. Depending on the application an analysis of needs and costs has to be ,made and in some cases shortECC can have more advantages than AES.

An example case when the application of the shortECC could be interesting is the following one. In the scenario it is assumed that both shortECC and AES could be used in the trusted group environment and that the data to be secured is broadcast in the network. After receiving by the nodes it is decrypted in order to be further processed. The number of nodes in the network is larger than 50. The encryption using shortECC uses private/public key pair of the node

issuing the data and only the nodes having its public key are able to decrypt the data correctly. In case of AES all nodes have to have one and the same key for encryption and decryption purposes if the data is to be broadcast within the group.

The mote-class attack within the framework of the trusted group, which tries to compute the secret key of the node sending the encrypted data, requires solving the ECDLP on the sensor node (assuming that the malicious node has the valid shortECC parameters). Solving the ECDLP allows the malicious node to impersonate the one being actual issuer of the data. The tests performed have shown that solving ECDLP on the sensor node equipped with MSP430F5438A microcontroller needs about 24 hours. In case of AES, all nodes are able to impersonate the one sending the data right away. Willing to avoid this situation requires to use different secret keys for each pair of nodes in the trusted group. What causes 50 encryptions is the example network, making the total cost larger compared to one shortECC encryption.

4.4.6 shortECC versus asymmetric digital signatures: length of the security parameters

In [47], the authors show that the standard public key cryptography security approaches are computationally very expensive for the wireless sensor nodes. The shortECC proposes modification of the ECDSA algorithm not using any hash function in the computations, what reduces the length of the data, which is processed by the authenticating algorithm. The asymmetric approaches proposed in [20] and in [59] are based on the discrete logarithm problem in some finite field and require hash functions in the computations. Thus, using these protocols with reduced key lengths causes on the one hand that the brute force attack can be performed easier than in the case of shortECC and on the other hand the hash functions are not appropriate if the intention of the protocol is to operate on short numbers, e.g. 32-bit long ones. The asymmetric approach proposed in [7] does not require a hash function and is originally working on large integers. Reducing the security parameters causes also in this case that the brute force attack can be performed easier than in the case of shortECC.

4.4.7 shortECC versus AES-GCM: confidentiality, integrity and authenticity

Table 4.4 presents the results of comparison of authenticated encryption methods using shortECC and AES-GCM. The numbers of clock cycles presented in the table were computed as the average values of 1000 encryptions and decryptions performed for each of the methods. The computations were performed on the sensor node equipped with the MSP430F5438A microcontroller. Also in this case

Table 4.4: Comparison of shortECC and AES-GCM

Security Approach	**shortECC**	**AEC-GCM**
Secret Synchronization	no needed	needed
Unique components	no needed	unique nonce needed
Knowledge about payload length	no needed	needed
Public Key Length (uncompressed)	64 bits	-
Private Key Length	32 bits	128 bits
Payload Length	32 bits	32 bits
Clock Cycles for authenticated encryption	925000	36000
Clock Cycles for verification and decryption	424320	34000
Plaintext Length	32 bits	32 bits
Authenticated Ciphertext Length	98 bits	160 bits

Table 4.5: Comparison of existing security protocols

WSN protocol	Confidentiality	Authentication	Using shortECC reasonable?
LSec [52]	symmetric	asymmetric	no
TinySec [24]	CBC	MAC	no
SPINS [46]	symmetric	MAC	no
LEAP++ [32]	96-bit ECC	MAC	yes
Eschenauer [17]	none	asymmetric	no
Ning [41]	none	hash-based	no
Chae [31]	none	512-bit RSA	yes

the reduction of the lengths of the keys in shortECC is not sufficient to make the ECC based cryptography competitive to AES based authentication schemes, with respect to the computational effort. Only the length of the authenticated ciphertext is shorter in case of shortECC.

4.4.8 shortECC in state of the art WSN security protocols: applicability

This section discuses the possibilities of the application of shortECC in existing security solutions for wireless sensor networks. Table 4.5 provides a selection of state of the art protocols and the possibilities of substituting already used security means by shortECC.

The LSec approach proposed in [52] is used in clusters/groups. The group communication in clusters is secured by the private keys exchanged periodically by asymmetric means. The shortECC requires standard asymmetric means for exchange of security parameters and thus, is itself inappropriate for key exchange.

The TinySec approach proposed in [24] provides authenticated encryption and authentication, but uses symmetric cryptography for these purposes. Since shortECC is computationally more expensive as symmetric approaches, it cannot replace CBC and MAC used in SPINS.

In case of the SPINS approach [46] symmetric cryptography and MAC are used to provide confidentiality and data authentication. Application of shortECC would increase the computational efforts needed for these two operation.

The approach LEAP++ presented in [32] proposes using short key elliptic curve cryptography for exchanging the public keys in the encrypted form. The ECC-128 or ECC-96 can be used as lightweight cryptosystem for key exchange. Since the public keys need to be exchanged securely and should be unknown outside some group/cluster, this scenario fulfils the requirements of shortECC.

A digital signature algorithm is used in the approach proposed in [17]. It is used for the key distribution purposes what in case of shortECC requires standard ECC. Thus, shortECC authentication cannot be used in this approach.

The Message specific puzzle described in [41] provides weak authentication based on hash function. Since the computational effort needed by hash function is negligible in comparison to shortECC, the method proposed in this thesis is inappropriate for these weak authentication purposes.

In the method proposed in [31] the broadcast authentication is performed using the asymmetric RSA/Rabin signature with message recovery, with security based on hardness of factorization of large integers. In this approach the RSA security parameters are changed periodically in order to make the system secure, because factorization of 512-bit RSA moduli can be performed within few hours.

This thesis shows that short key elliptic curve systems applied under specified conditions are secure for a longer period of time compared to the 512-bit RSA. Thus, substitution of RSA/Rabin signature by the one proposed in shortECC is possible, because shortECC is also designed to secure broadcast communication within the trusted group. Additionally, replacing the 512.bit RSA with 32-bit shortECC makes the approach feasible for WSN, where the still relatively large data blocks (512 bit) would cause high transmission costs.

Chapter 5

lmRNG: Pseudo-random Number Generator

This chapter presents the means for generating pseudo-random numbers that on one hand are suitable for cryptographic purposes and, on the other hand, are applicable in the resource constrained world of WSN [54]. The proposed generator was developed as a part of the shortECC approach, that requires pseudo-random numbers in the encryption and the digital signature protocols. Nevertheless, it can be used as an independent tool and its applicability is not restricted to the Wireless Sensor Networks only. The following section describes the concept and the overall idea of this approach and is followed by the sections introducing the detailed description of the two parts the generator consists of: the first one responsible for obtaining the seed - an initial value for the subsequent computations, and the second one, which is a deterministic algorithm outputting the pseudo-random numbers. The chapter includes the evaluation of the randomness and of the cryptographic applicability of the lmRNG using the NIST testing mechanisms and concludes with the comparative evaluation of the proposed approach and other solutions used in WSN environments.

5.1 Overview

Random number generators (RNG) can be classified as follows:

- True random number generators (TRNGs). This type of generators uses a non-deterministic source (entropy source) and some processing function (entropy derivation process) to produce random numbers. As an entropy source a device or some process generating unpredictable outputs can be

taken, e.g., digital signals from oscillators, mouse movements or noises in an electrical circuit.

- Deterministic random number generators (DRNGs). A DRNG produces pseudo-random numbers using some algorithm requiring one or more randomly selected input values - the seed. Since the outputs of the DRNG are generated by deterministic functions based on the seed it is very important that the seed is random and unpredictable. Thus, the seeds for DRNG should be generated using TRNG [43].

Following the guidelines by NIST the DNRG algorithm developed within the framework of this thesis consists of two blocks, the first one responsible for generating the seed, and the second one responsible for generating the pseudo-random numbers using the seed as its initial value. The algorithm was implemented and tested on wireless sensor nodes. The test platform was the IHPNode [23] which is equipped with the MSP4305438A microcontroller [55] with a built in ADC converter. The ADC converter changes the analog signals (continuously changing quantities) into digital data and the noise of this device was used to produce seeds.

5.2 Generation of the Seed

According to NIST, a DRNG requires an initial value-seed that is determined by an entropy source [1]. The seed used as the DRNG initial value has to contain sufficient entropy to assure that the outputs of the DRNG will be unpredictable. Entropy is defined as follows [2]:

Definition 14. *The entropy of a discrete random variable X is the expected amount of information that will be provided by an observation of X. In case when the information content is measured in bits and the expected information content of X is m bits, we say that the random variable X has m bits of entropy. The entropy $H(X)$ of discrete random variable $X = \{x_1, ..., x_n\}$ is computed as follows:*

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i), \tag{5.1}$$

where $p(x_i)$ is the probability of x_i. The value of the variable X can be predicted prior to its observation. The larger the entropy of the variable X, the larger the uncertainty of predicting the value of X. The entropy can be measured using diverse formulas. One of them is called min-entropy. It is used as a worst-case measure of the uncertainty of predicting the value of X. The min-entropy m of the variable X means that the probability of observing any particular value of X is less or equal to 2^{-m}.

5.2.1 The Proposed Seed Generator

According to the NIST recommendation [43], a non-deterministic source for generating the seeds should be used for a pseudo-random number generator. In this work the ADC module available on the test sensor node platform is used for that purpose. The exact source is the noise of the ADC module. In the test setup the ADC module is fed with the on-chip temperature sensor and this integrated hardware combination is used to generate the seed. The seed is generated by taking the least significant bit of each measurement, because it is the most sensitive bit, i.e., even small noise or oscillation of the temperature influences its value. These bits are concatenated in order to generate the seed of the required length.

5.2.2 Entropy Source Evaluation using NIST Test Suite

The special publication by NIST [2] describes the requirements for entropy sources and also the tests that shall be used for validation of these sources. The validation of the entropy sources is based on the correctness of the estimation of the min-entropy. The tests implemented in Python programming language are available from NIST upon request and were used within the framework of this work to evaluate the proposed source of the seed value. The following steps were performed in order to test the entropy source:

1. Generation of the input data for the test suite, i.e., a dataset of 1 000 000 1-bit samples, which are unprocessed.

2. Test if the input data is Independent and Identically Distributed (IID), i.e. *if it is a sequence of random variables where each element of the sequence has the same probability distribution and the other values and all values are mutually independent*, [2]. The claim that the input data is IID is required for full entropy sources, working in laboratory environment and being under constant control. Within the framework of this thesis a non-IID entropy source is considered.

3. Perform five tests for non-IID data sets to obtain the estimate of the min-entropy.

    - The Collision Test: *measures the mean time to the first collision in a dataset. The goal of the collision statistic is to estimate the probability of the most-likely state, based on the collision times*, [2].
    - The Partial Collection Test: *computes the entropy of a dataset based on how many distinct values in the output space are observed; it yields low entropy estimates for output streams that contain a small number of distinct output values, and yields high entropy estimates for output streams that diversify quickly*, [2].

- The Markov Test: *measures the dependencies between consecutive outputs from the noise source*, [2].
- The Compression Test: *computes the entropy rate of a dataset, based on how much the dataset can be compressed*, [2].
- The Frequency Test: *computes entropy based on the occurrence of the most-likely sample value*, [2].

4. After estimation of the entropy, the data set is subjected to the sanity checks: *tests that discover major failures in the design and gross overestimates of the entropy by the test suite. Failure to pass these tests means that the entropy source fails testing*, [2].

Evaluation Results

The NIST tests or non-IID entropy sources were used to evaluate 1000 sequences each 1000000 bit long (Figure 5.1). The min-entropy values determined for each sequence were between 0.83 and 0.92, thus the 0.83 value was taken as the min-entropy for the ADC with the temperature sensor as the entropy source.

5.3 Deterministic Pseudo-Random Number Generator

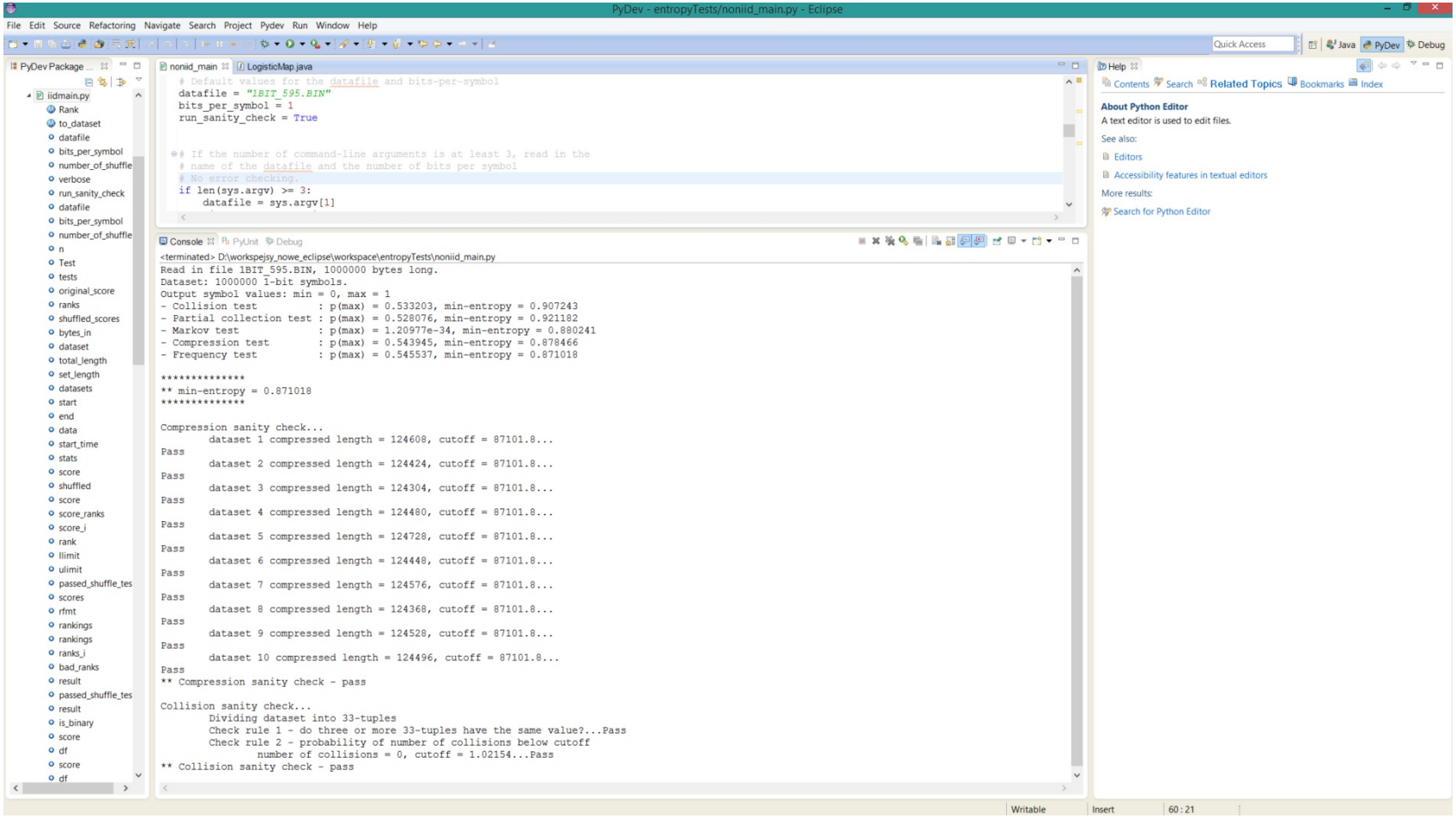
The deterministic algorithm generating the pseudo-random numbers for cryptographic purposes should assure that its outputs are unpredictable under the assumption that the seed is always kept secret. The outputs generated by the deterministic algorithms are in a form of binary sequences. It should be impossible to predict the next outputs in the generated sequence on the basis of any previously generated and known number. Also the backward discovery of the numbers in the sequence should be infeasible. The proposed approach was developed based on the properties and behaviour of the chaotic functions, which are briefly introduced in the following subsection.

5.3.1 Dynamical Systems and Mathematical Chaos

Mathematical chaos is a property of non-linear difference equations showing a large sensitivity of the solutions of these equations to small changes in their parameters. The features of the mathematical chaos can be used to produce non-deterministic algorithms for generation of pseudo-random numbers. A dynamical system is a set of coupled difference equations which determine how the state of the system evolves over time [27]. A non-linear difference equation, where the time is integer-valued is *a map* of the following form:

$$X_{t+1} = F(X_t). \tag{5.2}$$

Figure 5.1: NIST tests for entropy sources



For some parameters the dynamical system can exhibit chaotic behaviour. Such a system is sensitive to initial conditions, i.e., small changes of the input parameter cause significant differences in the future values.

Logistic Map

Logistic Map is a chaotic map representing a population model and it was introduced in [36]. It is a recurrence relation of degree two, mathematically written as:

$$x_{n+1} = rx_n(1 - x_n) \tag{5.3}$$

where the real number r is a control parameter and the initial element x_0 needs to be defined prior to the computations. The elements $x_0, x_1x_2, ...$ are real numbers lying in the interval $(0, 1)$. The Logistic Map is sensitive to the parameter r, which has to be chosen carefully. There are some values of r resulting in non-chaotic behaviour of the Logistic Map, e.g., according to [27] for r values below 3.57 the chaotic map is in periodic regime.

5.3.2 The Proposed Pseudo-Random Number Generator

Due to its simplicity and chaotic behaviour, the Logistic Map is a perfect candidate for a function generating random bits [35], [28]. The main disadvantage of the Logistic Map is that it operates on real numbers, what makes it computationally expensive and not suited for resource constrained devices. Computations on integers are performed faster than those performed on real numbers. Table 5.1 presents the clock frequency independent results of the measured time needed by the microcontroller for computing the iterations of the Logistic Map using different data types, i.e., the float numbers as it is used in the original form of the Logistic Map and for integers as proposed in the modification within this thesis. It also shows that even without the support of the hardware multiplier available on the test platform, the computations on the integers need only the half of the clock cycles, compared to those performed on floats. The hardware multiplier (HM) allows for six times faster integer computations compared to floats. And the presented numbers of clock cycles required for integer multiplication can be optimized further, since the multiplication result is divided modulo 2^{32} and thus only a partial 32-bit multiplication is needed.

Additionally, on most microcontrollers, and on the used microcontroller as well, the representation of floating point numbers has limited precision. On the test platform it is restricted to 7 digit precision, what can cause periodicity and lead to loss of chaotic behaviour of the Logistic Map. Thus, instead of the floating point operations, in the lmRNG approach a modified Logistic Map with integers is used and it is given by the following equation:

$$x_{n+1} = (x_n(1 + x_n) \mod 2^{32}) + 1 \tag{5.4}$$

Table 5.1: Clock cycles needed for Logistic Map iterations performed on MSP430F5438A

Data Type	**1 Iteration**	**32 Iterations**
float	522	16165
uint32	282	8485
uint32+HM	83	2117

And since the computations are performed on the integers greater than 0, the following special case has to be considered:

$$\text{if} \quad (x_n(1+x_n) \mod 2^{32}) + 1 = 0 \quad \text{then} \quad x_{n+1} = 1. \tag{5.5}$$

In order to reduce the number of multiplications in a single iteration, and thus to make the approach less computationally expensive, the parameter r used in the original Logistic Map is in the proposed modification, equal to 1. The modulo operation is required in order to ensure that the outputs of the algorithm will be in range $0, ..., 2^{32}$ what is required by the shortECC approach. Independently from the operations used in the modified formula, performing the modulo operation at the end of computations of the single iteration and incrementing it by one causes that the result is always greater than 0. Thus, the originally used subtraction operation was changed by addition.

The here proposed pseudo-random number generator uses two so modified Logistic Maps that require two different seeds as inputs.

$$x_{n+1} = x_n(1+x_n) \mod 2^{32} + 1 \tag{5.6}$$

$$y_{n+1} = y_n(1+y_n) \mod 2^{32} + 1 \tag{5.7}$$

In order to generate a n-bit long number z, n iterations of both maps are performed. In each iteration the outputs of both maps are compared and on the basis of this comparison the i-th bit of the number z is generated as follows:

$$\text{if} \quad (x_i > y_i) \text{ then} \quad z_i = 1, \text{ otherwise} \quad z_i = 0. \tag{5.8}$$

The performed experiments have shown that using only a single modified Logistic Map to reduce computations and generating the bit streams by taking from the iteration results different combinations of bits, caused the generated bit sequences to have poor quality and, as a result, to fail the NIST tests for randomness sources. Additionally, the final method for determining the value of the i-th bit was proposed after a number of experiments. The combination of two maps with the decision based on the result of the comparison of their outputs causes the approach to have the desired properties, i.e., the output is random. Further,

using two maps, each using a 32-bit seed as input increases the number of possible PRNG sequences. This reduces the chance of a brute force attack to recover the seeds on the basis of the generated numbers.

5.3.3 Randomness Evaluation using NIST Test Suite

The special publication of NIST [43] provides 15 statistical tests that shall be applied to a binary sequence to perform the evaluation and comparison of it to a truly random sequence. The tests examine the sequence looking for some defined sub-sequences or patterns whose presence means that the sequence is non-random. In this thesis the NIST testing suite was used to test the randomness of the binary sequences produced by the proposed deterministic solution. For the randomness assessment of the proposed approach 1000 binary sequences, each 1000000-bit long, were generated.

The NIST testing suite provides a set of statistical tests assessing the randomness of an evaluated sequence. The sequence consists of zeros and ones that represent the binary form of the numbers. The aim of a statistical test is to assess the probability that the null hypothesis (H0) is fulfilled, where H0 means that the sequence is random. Each test is focused on checking a different property of the sequence in order to test the H0 and either accepts or rejects it. For this purpose in each test the randomness statistic is used. The randomness statistic has a distribution of possible values, and mathematical methods determine the reference distribution for it. The reference distribution allows computing a critical value, which determines the boundary between those samples resulting in a randomness statistic that leads to rejecting the H0 and those that lead to accepting the H0. For a given sequence the test statistic value is computed and if it is bigger than the critical value, the H0 is rejected. On the basis of the test statistic the P_{values} is computed. The method for computing the P_{values} is chosen for each test independently and a special function is used for this purpose. It can be for example the error function or the gamma function. It is the probability, that a perfect random number generator would have produced a sequence less random than the sequence that was tested. If the $P_{values} = 1$ then the sequence appears to be random. For the NIST test suite the P_{values} has to be larger than 0.01 to accept the null hypothesis. The number $\alpha = 0.01$ is called the significance level.

The following tests are part of the NIST Testing Suite:

- Frequency (Monobit) Test: checks if the proportion of zeros and ones in the sequence is about the same and compares it to the values expected from the truly random sequence. The next tests depend on passing this test.

- Frequency Test within a Block: checks if the proportion of ones in the M-bit block is about M/2, what is expected from truly random sequence.

- Runs Test: determines the number of runs in the sequence, where a run is a subsequence consisting of identical bits and bounded with bits of opposite value. The number of runs having various lengths is compared with values expected from truly random sequence.

- Test for the Longest Run of Ones in a Block: checks if the length of the longest run of ones in the sequence is compared to the length of the runs of ones in the truly random sequence.

- Binary Matrix Rank Test: *checks for linear dependence among fixed length sub-strings of the original sequence, [43].*

- Discrete Fourier Transform (Spectral) Test: detects the periodicities (peaks), e.g. repetition of patterns that are near to each other.

- Non-overlapping Template Matching Test: detects if the sequence contains too many given non-periodic m-bit patterns. For the detection the m-bit window corresponding to the non-periodic pattern is used. The window slides bit by bit as long as the pattern is found and after that the window is reset to the bit after the pattern and the search resumes.

- Overlapping Template Matching Test: determines the number of occurrences of the given pattern. This test also uses m-bit window to find the pattern and the widow slides bit by bit as long as the pattern is found. After that the window slides one bit and the search resumes.

- Maurer's Universal Statistical Test: computes the number of bits between the same patterns, where the number is the measure related to the length of the sequence after compression. The test detects if the sequence can be compressed without loss of information, what implies the sequence to be non-random.

- Linear Complexity Test: measures the length if a linear feedback shift register. When the computed value is too short it implies non-randomness of the sequence.

- Serial Test: checks the frequency of all possible m-bit patterns in the sequence. If the probability that a given m-bit sequence is comparable with the probabilities of other m-bit patterns, the sequence appears to be random.

- Approximate Entropy Test: *compares the frequency of overlapping blocks of two consecutive/adjacent lengths (m and m+1) against the expected result for a random sequence, [43].*

- Cumulative Sums (Cusum) Test: computes the cumulative sum of adjusted digits in the sequence, using $(-1, +1)$. This value is then compared to the one expected from the random sequence. *The cumulative sum may be considered as a random walk. For a random sequence, the excursions of the random walk should be near zero, [43]*

- Random Excursion Test: computes the number of cycles with K visits in cumulative sum random walk. *A cycle of a random walk consists of a sequence of steps of unit length taken at random that begin at and return to the origin. The purpose of this test is to determine if the number of visits to a particular state within a cycle deviates from what one would expect for a random sequence, [43]*

- Random Excursion Variant Test: *detect deviations from the expected number of visits to various states in the random walk, [43]*

After the binary sequences have been tested, there are two approaches for interpreting the empirical test results. In the first approach the proportion of the sequences that passed the test, i.e. having $P_{values} > 0.01$ is computed. The proportion is computed on the basis of the number of sequences to be tested and the sequences that passed the test. Let 1000 be the first number, and 998 the second one. Then the proportion is equal to 0.998 and the range of acceptable proportions is computed using the formula:

$$(1 - \alpha) \pm 3\sqrt{\frac{(1 - \alpha)\alpha}{m}} \tag{5.9}$$

where m is the sample size and α is the significance level. Thus, for 1000 sequences the proportion should be larger than 0.9805607. In the second approach the distribution of the P_{values} is determined in order to check the uniformity. For that the $P_{Uvalues}$ of all the P_{values} is computed. The $P_{Uvalues}$ is computed using the special function as defined in the NIST suite. If $P_{Uvalues} \leq 0.0001$, then the P_{values} can be considered to be uniformly distributed.

Evaluation Results

Table 5.2 and Table 5.3 present the results of NIST tests performed on the random bit streams generated by the proposed pseudo-random number generator. For the 1000 tested sequences, the proportion of passed tests should be larger than 0.9805607. All the tests were passed, what means that proposed approach generates pseudo-random numbers.

Table 5.2: Results of Non-Parametrized Tests.

Statistical Test	**Distribution of P_{values}**	**Proportion of passed tests**
Frequency (Monobit) Test	0.500279	0.9910
Cumulative Sums Test		
forward sums	0.668321	0.9920
reverse sums	0.340858	0.9920
Runs Test	0.299736	0.9880
Test for the Longest Run of Ones in a Block	0.914025	0.9850
The Binary Matrix Rank Test	0.834308	0.9950
The Discrete Fourier Transform (Spectral) Test	0.254411	0.9890
Random Excursions Test		
x = -4	0.137669	0.9869
x = -3	0.287113	0.9869
x = -2	0.306059	0.9918
x = -1	0.613238	0.9918
x = 1	0.770642	0.9886
x = 2	0.825651	0.9902
x = 3	0.901436	0.9837
x = 4	0.256024	0.9806
Random Excursions Variant Test		
x = -9	0.856548	0.9935
x = -8	0.899148	0.9951
x = -5	0.487035	0.9918
x = -2	0.552416	0.9821
x = 2	0.506324	0.9886
x = 5	0.275709	0.9951
x = 7	0.041549	0.9886
x = 9	0.773817	0.9886

Table 5.3: Results of Parametrized Tests.

Statistical Test	**Distribution of P-Values**	**Proportion of passed tests**
Linear Complexity Test (block length 500)	0.452173	0.9910
Serial Test (block length 16)	0.208837	0.9870
Serial Test (block length 14)	0.647530	0.9850
Non-Overlapping Template Matching Test		
template = 000110111	0.435430	0.9920
template = 001001101	0.191687	0.9940
template = 001011011	0.595549	0.9910
template = 101111100	0.755819	0.9850
template = 110111000	0.395940	0.9920
template = 111010100	0.325206	0.9930
template = 111101100	0.204439	0.9900
Overlapping Template Matching Test	0.672470	0.9900
Maurer's Universal Statistical Test	0.508172	0.9870
Frequency Test Within a Block (block size 10^4)	0.666245	0.9870
Approximate Entropy Test (block length 10 bits)	0.429923	0.9870

5.4 Evaluation of the Proposed Approach

Table 5.4 presents the comparison of the proposed approach with the currently used solutions for generating random numbers on low power devices. The table is divided into three parts: first one (PRNG) specifies the method producing the outputs considered as random numbers, the second one (Seed) names the entropy source serving as the input for the PRNG methods and the third one defines if there is a necessity of cooperation between sensor nodes in order to generate random numbers.

- The PRNG part contains the information about the algorithms used for generating the random numbers, about using the additional hardware supporting the algorithms and about the results of the tests performed in order to check if the outputs of the PRNG methods can be considered as pseudo-random numbers.
  - Algorithm Used - There are various algorithms used to generate the final outputs in the compared methods. Three methods [18], [33] and [19] use MAC based functions, two others [58] and [51] use operations on bits, e.g. XOR. The approach presented in this thesis uses chaos based Logistic Map as the deterministic source of randomness.

Table 5.4: Comparison with State of the Art proposals

		lmRNG	**Francillon [18]**	**Wang [58]**	**Seetharam [51]**	**Lo Re [33]**	**Gaglio [19]**
PRNG	**Algorithm Used**	Logistic Map	CBC-MAC	Bit skipping, Bit Counting	XOR	CMAC	MAC
	Hardware Involved	none	radio	single electron transistor	timer, radio	none	none
	NIST Randomness Tests	passed	not performed	failed	not performed	not performed	passed
Seed	**Seed Source**	ADC (LSB)	given at programming time	telegraphic signal	node ID	ADC	sensors
	Hardware Involved	ADC	radio	single electron transistor	timer, radio	ADC	sensors
	Seed Entropy	0.83	assumed to be enough	-	-	-	0.57, 0,71, 0,66
	NIST Entropy Source Tests	passed	not performed	not performed	not performed	not performed	not performed
Single Node Generator		yes	needs communicating nodes	yes	needs communicating nodes	needs communicating nodes	needs communicating nodes

  - Hardware Involved - The PRNG parts require the hardware modules in following cases: in [18] the radio is used to generate the input values for the CBC-MAC, in [58] the telegraphic signals from single electron transistor are taken as inputs for the bit skipping or bit counting operations, in [51] the timer values are XORed with seed which in turn is constantly updated using the check sums of radio packets.
  - NIST Randomness Tests - Three methods were tested using the NIST Test Suite [43] for pseudo-random numbers. For the algorithm proposed in this thesis 1000 sequences each containing 1 000 000 bits were generated and tested. All the sequences passed the tests. Another approach tested using the NIST Test Suite is the one presented in [58]. There is no information about the number of sequences that were tested and the presented results show that not all the tests were passed, e.g. the Runs Test. The third algorithm tested is the one presented in [19]. The authors performed the tests for 100 sequences and all the tests were passed. The remaining three methods were not tested using the NIST Test Suite.

- The Seed part contains the characteristics of the seeds used as input values for the above described PRNG methods. There is information about the source of the input values, about the entropy of the input values and about the results of the tests performed in order to check if the values have enough entropy to be considered as inputs for PRNG used for cryptographic purposes.

  - Seed Source - The proposed approach uses the least significant bits of the values from the ADC temperature sensor in order to produce seeds. Also in case of the solution presented in [33], the ADC values are used as seed for the generation algorithm. During the research on the approach presented in this thesis, the raw values generated by the ADC were tested using the NIST tests for entropy sources and the tests were failed. The method presented in [18] requires that the initial value is given at the programming time and stored in the internal EEPROM. While running the generating algorithm, the seed is changed and the new value is saved in the memory. According to the authors this operation is infrequent. The initial values for the approach presented in [58] are obtained using the single electron capture/emission process. The algorithm used in [51] uses sensor node ID as the input value, what causes that, in case of reboot for example, the initial value is always the same. The last method in comparison, presented in [19] uses the sensor measurements as the seed sources.

  - Hardware Involved - Only the approach from [51] does not require hardware as entropy source. The remaining methods use on-board entropy sources: the approach presented in this work and [33] use ADC. The approach presented in [19] uses sensor measurements received from the neighbouring sensor nodes. The method proposed in [18] requires radio, because the reseeding operation is done using data from packets received by radio. The initial values in [58] require the single electron transistor existing on the sensor board.
  - Seed entropy - Only two approaches give detailed information about the measured entropy of the seed produced by the used entropy sources. Both cases consider the one bit samples and for such a case the full entropy value is equal to 1, what, following Definition **??**, gives 50% chance to guess the value of the next bit sample. The seed source chosen for the generator presented in this thesis has an entropy equal to 0.83, what gives 56% chance to guess the value of the one bit sample. The entropy was measured using the NIST tests for entropy sources [2]. The second method providing the information about entropy is the one presented in [19]. The results given 0.57, 0.66 and 0.71 are worse than the result of the proposed approach, because the probabilities of guessing the bit values in one bit samples are 67%, 63% and 58%, respectively.
  - NIST Entropy Source Tests - The only method tested using the NIST entropy source tests is the one proposed in this work. The rest of the approaches do not provide any information about checking if the entropy sources they use are good enough to be used in cryptographic pseudo-random number generators.

- The Single Node Generator part - The approach presented in this thesis and the one presented in [58] do not require any cooperation between sensor nodes in order to generate the pseudo-random numbers. The three remaining methods need that the nodes communicate in order to produce the pseudo-random numbers, thus the radio communication is directly involved in their generation processes.

lmRNG versus tinyRNG

Table 5.5 presents the comparison of the proposed approach with TinyRNG with respect to the time needed for different steps of the algorithm. The lmRNG generator does not need any initialization, what in case of TinyRNG takes about 147 milliseconds. In case of TinyRNG the generation of seeds is preceded by a number of entropy accumulations, each taking about 2 milliseconds. The generation of TinyRNG seed itself takes about twice the time needed by the proposed lmRNG

Table 5.5: Comparison with HM vs TinyRNG @ 8MHz

Operation	**lmRNG**	**TinyRNG**
Initialization	0	146 ms
Seed Generation	530 μs	1.13 ms
Entropy Accumulation	0	2.16 ms
Generation of 64 bits	1 ms	440 μs

generator. And the generation of 64 random bits takes about 1 millisecond by the here proposed generator, compared to about 440 microseconds in TinyRNG. Altogether, the whole process of generating the 64 random bits takes less time in case of the here proposed approach, even if the initialization phase is not taken into account. Additionally, the main advantage of the proposed approach is that it allows for generation of random numbers also in case when the radio transceiver is turned off, e.g., the sensor node is in power saving mode. This is impossible in case of TinyRNG. Keeping the radio to be turned off as often as possible is one of the main requirements for the wireless sensor networks.

Chapter 6

Conclusions

This chapter concludes the thesis. It provides a short summary of the results of the main investigation - analysis of the applicability of elliptic curve cryptography with short parameters. But it also presents the contributions of the thesis and sketches (possible and planned) future developments based on its results.

6.1 Summary

This thesis investigated the applicability of short key asymmetric cryptography in low power Wireless Sensor Networks. The decision about choosing the ECC as the base for the proposed approach and the investigation was caused by the fact that elliptic curve cryptography provides the same security level as public key cryptography based on the modular arithmetic, while using much shorter lengths of the keys [3]. In order to start the further investigation and to propose the approach for ECC with reduced key lengths, it was necessary to recognize the constraints of WSN influencing the effectiveness of computations when performing the cryptographic algorithms. Another factor that influenced the target key lengths was the size of the data that is usually processed in the WSN applications. After considering the above mentioned issues it was decided to focus on 32-bit keys for the ECC based cryptosystem using these key lengths, named shortECC. Under standard conditions for the ECC applications, such key lengths are considered to be insecure [3]. Thus, an application scenario, together with its constraints and requirements, was defined for the new approach. The main restriction here was that the shortECC approach can be only used in a closed group of nodes and that all the shortECC parameters are shared only with the members of this group.

Within the closed group of nodes, it was intended to use the shortECC cryptosystem for provision of confidentiality and authentication. The ElGamal ECC

encryption and ECDSA were chosen as the base algorithms ensuring these two security requirements. Further, it was important to choose elliptic curves of a specified order. It was shown that due to the immunity to the Pohlig-Helmann attack and simultaneously due to the provision of ambiguity of correct results, while performing the brute force attack, the elliptic curves of prime order are good candidates to be used in shortECC.

The next step in the investigation was the selection of sub-components required by the cryptographic algorithms, like a hash function or a pseudo-random number generator. It was shown that the state of the art solutions either do not operate on such short numbers as these needed by shortECC, or are too expensive (or complex) to be applicable. Hence, in order to omit the necessity of using hash functions a modified version of ECDSA was proposed within the framework of this thesis. The provision of the necessary cryptographic pseudo-random numbers for shortECC is ensured by the novel algorithm presented and evaluated in this thesis - lmRNG [54].

6.2 Contributions

Considering the constraints and the requirements of the shortECC approach it was apparent that the state of the art ECC as well as the pseudo-random number generators are computationally too expensive to be used in short key cryptographic algorithms. Thus, new methods and approaches providing above mentioned functionalities were proposed.

Lightweight Security Approach

For the ECC the new approach involves modifications of the cryptosystem definition and algorithms. As already mentioned, shortECC changes the distribution of parameters between the secret and public domain. Besides this change, most of the standard ECC algorithms can be applied.

The motivation for providing the modified authentication scheme was that the state of the art approaches use hash functions that produce outputs that are not applicable in the shortECC approach. Further, even if a hash function generating output with the needed size would be available, its application here would be pointless due to high probability of collisions. Thus, an algorithm that does not require a hash function was necessary. The proposed algorithm is a modified version of the standard Elliptic Curves Digital Signature Algorithm (ECDSA). It is a 1-to-1 digital signature algorithm without appendix with message recovery. The security of the proposed protocol is guaranteed under the condition of keeping the shortECC parameters secret.

Cryptographic Pseudo-random Number Generator

The motivation for the research on a new approach for generating cryptographically secure pseudo-random numbers was the fact that the sub-components of shortECC need to be computationally inexpensive and minimal energy consuming and none of the state of the art solutions fulfils these requirements. The proposed novel cryptographic pseudo-random number generator uses the properties of mathematical chaos for the provision of unpredictable pseudo-random numbers. The algorithm is based on the logistic map - a chaotic function - which was modified in order to be applicable in WSN. Instead of real numbers it operates on integers, since integer arithmetic is more natural for sensor nodes. The generator is able to generate random numbers of an arbitrary length, from single or several bits to thousands or even millions of bits. The outputs produced by the new algorithm were tested according to the NIST recommendations for deterministic pseudo-random algorithms and entropy sources. All the test defined by NIST were passed, what indicates that the proposed pseudo-random number generator can be used for cryptographic purposes.

6.3 Results of the Applicability Investigation

To summarize the applicability analysis the following statement can be defined. The evaluation of the security of shortECC has indicated that the approach is secure only if the shortECC parameters are shared by members of a closed and trusted group of nodes and only if the points on the shortECC elliptic curves, being results of the algorithms, are exchanged in the compressed form.

The shortECC approach was compared to symmetric and asymmetric state of the art solutions. It was shown that it works slower than the symmetric AES encryption, but is significantly faster than the standard ECC algorithms, especially when the bit-length of the data to be secured does not exceed 32.

The shortECC approach could be used to substitute the Elliptic Curves Integrated Encryption Scheme (ECIES), which provides the encryption and decryption of data, but besides the ECC algorithms, it also requires symmetric encryption and hash function. The ECIES scheme operates on standard 128-bit ECC what allows computations on the 128-bit (or larger) outputs of the AES and of the hashing functions.

Within the framework of this thesis it was also shown that the shortECC cryptosystem can replace cryptographic solutions in the state of the art cryptographic protocols. For instance, in LEAP++ presented in [32] it is possible to use the short key elliptic curve cryptography after fulfilling the shortECC requirements. Further, according to the requirements defined in the scenario for the RSA/Rabin signature [31] it is feasible to substitute it by the one proposed for shortECC.

6.4 Future Work

This thesis was (partially) realized within the European research project SMARTIE [11] and the proposed solutions are going to be applied in this project. But besides that, the results of this thesis are also going to be used and further investigated in several other activities. Thus, the next steps on the research on security for Wireless Sensor Networks based on the results of this thesis are as follows.

In order to investigate the applicability of other strong security approaches in low power environments, it will be researched if cryptography based on hyperelliptic curves (HECC) better suits the WSN applications. These curves can provide the same level of security as standard elliptic curves, while using significantly shorter parameters, i.e. they operate on 50 - 80 bit long numbers [40]. This can help to achieve the goal of reducing the amount of the exchanged data. The computational expensiveness of the algorithms operating on hyperelliptic curves is comparable to the expensiveness of the standard ECC approaches [40]. It will be challenging to adjust the HECC algorithms to be less computationally expensive and thus applicable for WSN.

Another further research based on the results from this thesis involves the lmRNG. It will be investigated if the random numbers generated by the proposed cryptographic pseudo-random number generator can be used as a key chain. There are cryptographic algorithms that are lightweight, but cause a huge burden for managing and synchronizing the credentials. One of these algorithms is the CaMyTs approach [5] providing the symmetric homomorphic encryption that allows to aggregate encrypted values. This approach will be used together with the lmRNG as the key chain in the European project e-balance to process the protect the private data of the energy grid users.

Bibliography

[1] Elaine Barker and John Kelsey. Nist draft special publication 800-90a. recommendation for random number generation using deterministic random bit generators, 2012. [cited at p. 72]

[2] Elaine Barker and John Kelsey. Nist draft special publication 800-90b. recommendation for the entropy sources used for random bit generation, 2012. [cited at p. 4, 72, 73, 74, 85]

[3] Elaine Barker and Dang Quynh. Special publication 800-57, part 3, revision 1, 2015. [cited at p. 1, 2, 10, 87]

[4] Ian F Blake, Gadiel Seroussi, and Nigel Smart. *Elliptic curves in cryptography*, volume 265. Cambridge university press, 1999. [cited at p. 13, 14, 48, 49, 101]

[5] Claude Castelluccia, Einar Mykletun, and Gene Tsudik. Efficient aggregation of encrypted data in wireless sensor networks. In *Mobile and Ubiquitous Systems: Networking and Services, 2005. MobiQuitous 2005. The Second Annual International Conference on*, pages 109–117. IEEE, 2005. [cited at p. 90]

[6] Haowen Chan, Adrian Perrig, and Dawn Song. Random key predistribution schemes for sensor networks. *Security and Privacy, IEEE Symposium on*, 0:197, 2003. [cited at p. 39]

[7] Chin-Chen Chang and Ya-Fen Chang. Signing a digital signature without using one-way hash functions and message redundancy schemes. *Communications Letters, IEEE*, 8(8):485–487, 2004. [cited at p. 34, 67]

[8] Henri Cohen. *A course in computational algebraic number theory*, volume 138. Springer Science & Business Media, 2013. [cited at p. 14, 47]

[9] Henri Cohen and Gerhard Frey, editors. *Handbook of elliptic and hyperelliptic curve cryptography*. CRC Press, 2005. [cited at p. 8, 9, 12, 15, 18]

[10] Les composants associes. Prime i.t. `http://compoasso.free.fr/primelistweb/page/prime/liste_online_en.php`, 2015. [Online; accessed 06-May-2015]. [cited at p. 54]

[11] SMARTIE Consortium. Smartie secure and smarter cities data management, 2015. [cited at p. 90]

[12] Joan Daemen and Vincent Rijmen. *The design of Rijndael: AES — the Advanced Encryption Standard*. Springer-Verlag, 2002. [cited at p. 27]

[13] Whitfield Diffie and Martin E Hellman. New directions in cryptography. *Information Theory, IEEE Transactions on*, 22(6):644–654, 1976. [cited at p. 28]

[14] Morris Dworkin. Recomendation for block cipher modes of operation. nist special publication 800-38a, 2001. [cited at p. 17]

[15] Morris Dworkin. Recomendation for block cipher modes of operation: The ccm mode for authentication and confidentiality. nist special publication 800-38c, 2004. [cited at p. 34]

[16] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in Cryptology*, pages 10–18. Springer, 1985. [cited at p. 30]

[17] Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM conference on Computer and communications security*, CCS '02, pages 41–47, New York, NY, USA, 2002. ACM. [cited at p. 38, 68, 69]

[18] Aurélien Francillon and Claude Castelluccia. Tinyrng: A cryptographic random number generator for wireless sensors network nodes. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops, 2007. WiOpt 2007. 5th International Symposium on*, pages 1–7. IEEE, 2007. [cited at p. 41, 49, 82, 83, 84, 85]

[19] Vincenzo Gaglio, Alessandra De Paola, Marco Ortolani, and Giuseppe Lo Re. A trng exploiting multi-source physical data. In *Proceedings of the 6th ACM workshop on QoS and security for wireless and mobile networks*, pages 82–89. ACM, 2010. [cited at p. 42, 49, 82, 83, 84, 85]

[20] Marc Girault. Self-certified public keys. In *Advances in CryptologyEUROCRYPT91*, pages 490–497. Springer, 1991. [cited at p. 32, 67]

[21] Darrel Hankerson, Alfred J. Menezes, and Scott Vanstone. *Guide to Elliptic Curve Cryptography*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003. [cited at p. 3, 12, 13, 32, 33, 47, 53, 55, 101]

[22] Gerhard Jaeschke. On strong pseudoprimes to several bases. *Mathematics of Computation*, 61(204):915–926, 1993. [cited at p. 49]

[23] P. Langendoerfer K. Piotrowski, A. Sojka. Body area network for first responders - a case study. In *The 5th International Conference on Body Area Networks, BodyNets*. ACM, 2010. [cited at p. 72]

[24] Chris Karlof, Naveen Sastry, and David Wagner. Tinysec: a link layer security architecture for wireless sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, SenSys '04, pages 162–175, New York, NY, USA, 2004. ACM. [cited at p. 36, 68, 69]

[25] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of computation*, 48(177):203–209, 1987. [cited at p. 4, 29, 30, 47, 50]

[26] Neal Koblitz. *Algebraic aspects of cryptography*, volume 3. Springer Science & Business Media, 2012. [cited at p. 48, 50]

[27] Ljupco Kocarev and Shiguo Lian. *Chaos-based cryptography*. Springer, 2011. [cited at p. 74, 76]

[28] S. Lian L. Kocarev. *Chaos-based Cryptography*, volume 354. Springer, 2011. [cited at p. 76]

[29] Arjen K Lenstra and Eric R Verheul. Selecting cryptographic key sizes. *Journal of cryptology*, 14(4):255–293, 2001. [cited at p. 64]

[30] Hendrik W Lenstra Jr. Factoring integers with elliptic curves. *Annals of mathematics*, pages 649–673, 1987. [cited at p. 54]

[31] Chae Lim. Practical broadcast authentication using short-lived signatures in wsns. In Heung Youm and Moti Yung, editors, *Information Security Applications*, volume 5932 of *Lecture Notes in Computer Science*, pages 366–383. Springer Berlin / Heidelberg, 2009. [cited at p. 40, 68, 69, 89]

[32] Chae Hoon Lim. Leap++: A robust key establishment scheme for wireless sensor networks. In *Distributed Computing Systems Workshops, 2008. ICDCS '08. 28th International Conference on*, pages 376 –381, June 2008. [cited at p. 38, 68, 69, 89]

[33] Giuseppe Lo Re, Fabrizio Milazzo, and Marco Ortolani. Secure random number generation in wireless sensor networks. *Concurrency and Computation: Practice and Experience*, 2014. [cited at p. 42, 82, 83, 84, 85]

[34] V Gayoso Martínez, L Hernández Encinas, and C Sánchez Ávila. A survey of the elliptic curve integrated encryption scheme. *ratio*, 80(1024):160–223, 2010. [cited at p. 31]

[35] R. May. *Theoretical ecology : principles and applications / edited by Robert M. May.* Blackwell Scientific, Oxford, 1976. [cited at p. 76]

[36] Robert M May et al. Simple mathematical models with very complicated dynamics. *Nature*, 261(5560):459–467, 1976. [cited at p. 76]

[37] David A McGrew and John Viega. The security and performance of the galois/-counter mode (gcm) of operation. In *Progress in Cryptology-INDOCRYPT 2004*, pages 343–355. Springer, 2005. [cited at p. 35]

[38] Alfred J. Menezes, Scott A. Vanstone, and Paul C. Van Oorschot. *Handbook of Applied Cryptography*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1996. [cited at p. 8, 29, 30, 31, 54, 101]

[39] Victor S Miller. Use of elliptic curves in cryptography. In *Advances in Cryptology-CRYPTO85 Proceedings*, pages 417–426. Springer, 1986. [cited at p. 29]

[40] Debajyoti Mukhopadhyay, Ashay Shirwadkar, Pratik Gaikar, and Tanmay Agrawal. Securing the data in clouds with hyperelliptic curve cryptography. In *Information Technology (ICIT), 2014 International Conference on*, pages 201–205. IEEE, 2014. [cited at p. 90]

[41] Peng Ning, An Liu, and Wenliang Du. Mitigating dos attacks against broadcast authentication in wireless sensor networks. *ACM Trans. Sen. Netw.*, 4:1:1–1:35, February 2008. [cited at p. 39, 41, 68, 69]

[42] NIST. Specification for the advanced encryption standard (aes). Federal Information Processing Standards Publication 197, 2001. [cited at p. 27]

[43] NIST. A statistical test suite for random and pseudorandom number generators for cryptographic applications, 2010. [cited at p. 4, 42, 72, 73, 78, 79, 80, 84]

[44] NSA. Skipjack and kea algorithm specifications, 1998. [cited at p. 28]

[45] Adrian Perrig, Ran Canetti, J Doug Tygar, and Dawn Song. The tesla broadcast authentication protocol. *RSA CryptoBytes*, 5, 2005. [cited at p. 37]

[46] Adrian Perrig, Robert Szewczyk, J. D. Tygar, Victor Wen, and David E. Culler. Spins: security protocols for sensor networks. *Wirel. Netw.*, 8:521–534, September 2002. [cited at p. 36, 40, 68, 69]

[47] Krzysztof Piotrowski, Peter Langendoerfer, and Steffen Peter. How public key cryptography influences wireless sensor node lifetime. In *Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks*, pages 169–176. ACM, 2006. [cited at p. 1, 2, 67]

[48] Axel York Poschmann. Lightweight cryptography: cryptographic engineering for a pervasive world. In *Ph. D. Thesis*. Citeseer, 2009. [cited at p. 50]

[49] Ronald L. Rivest and Jacob C. N. Schuldt. Spritz—a spongy RC4-like stream cipher and hash function. Presented at Charles River Crypto Day (2014-10-24). [cited at p. 27]

[50] Ronald L Rivest, Adi Shamir, and Len Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978. [cited at p. 19, 30]

[51] D. Seetharam and Sokwoo Rhee. An efficient pseudo random number generator for low-power sensor networks [wireless networks]. In *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, pages 560–562, Nov 2004. [cited at p. 42, 82, 83, 84, 85]

[52] Riaz Ahmed Shaikh, Sungyoung Lee, Mohammad A. U. Khan, and Young Jae Song. Lsec: Lightweight security protocol for distributed wireless sensor network. In *PWC*, pages 367–377, 2006. [cited at p. 35, 68]

[53] Claude E Shannon. Communication theory of secrecy systems*. *Bell system technical journal*, 28(4):656–715, 1949. [cited at p. 17]

[54] Anna Sojka and Krzysztof Piotrowski. lmrfng: A lightweight pseudorandom number generator for wireless sensor networks. In *SECRYPT*, pages 358–363, 2012. [cited at p. 50, 71, 88]

[55] TI. Msp430x5xxx/msp430x6xxx family. user's guide. `http://www.ti.com/lit/ug/slau208i/slau208i.pdf`, 2010. [cited at p. 64, 72]

[56] Yuh-Min Tseng, Jinn-Ke Jan, and Hung-Yu Chien. Digital signature with message recovery using self-certified public keys and its variants. *Applied Mathematics and Computation*, 136(2):203–214, 2003. [cited at p. 32]

[57] Lawrence C Washington. *Elliptic curves: number theory and cryptography*. CRC press, 2008. [cited at p. 11, 51]

[58] F.B. Zhu Y.H. Wang, H.G. Zhang. An efficient random number generator for ad hoc sensor network. In *Wireless Communications, Networking and Mobile Computing, 2006. WiCOM 2006.International Conference on*, pages 1 –4, sept. 2006. [cited at p. 42, 82, 83, 84, 85]

[59] Fangguo Zhang, Willy Susilo, and Yi Mu. Identity-based partial message recovery signatures (or how to shorten id-based signatures). pages 45–56, 2005. [cited at p. 34, 67]

[60] Sencun Zhu, Sanjeev Setia, and Sushil Jajodia. Leap+: Efficient security mechanisms for large-scale distributed sensor networks. *ACM Trans. Sen. Netw.*, 2:500–528, November 2006. [cited at p. 37]

Used Abbreviations

ACK	Acknowledgement
AES	Advanced Encryption Standard
CBC	Cipher Clock Chaining
CBC-MAC	Cipher Clock Chaining Message Authentication Code
CCM	Counter with CBC-MAC mode
CFB	Cipher Feedback
CMAC	Cipher-based Message Authentication Code
CTR	Counter Mode
DES	Data Encryption Standard
DLP	Discrete Logarithm Problem
DRNG	Deterministic Random Number Generator
ECB	Electronic Codebook
ECC	Elliptic Curves Cryptography
ECDLP	Elliptic Curve Discrete Logarithm Problem
ECDSA	Elliptic Curve Digital Signature Algorithm
ECIES	Elliptic Curves Integrated Encryption Scheme
GCM	Galois/Counter Mode
HECC	Hyperelliptic Curves Cryptography
HMAC	keyed-Hash Message Authentication Code

HRNG	Hybrid Random Number Generator
IDS	Intrusion Detection System
IID	Independent and Identically Distributed
IV	Initialization Vector
MAC	Message Authentication Code
NIST	National Institute of Standards and Technology
OFB	Output Feedback
PKG	Private Key Generator
PKI	Public Key Infrastructure
RNG	Random Number Generator
TRNG	True Random Number Generator
WSN	Wireless Sensor Network

List of Figures

List of Algorithms

List of Tables